Modelos de uso y recomendación social en entornos de enseñanza

Javier Alonso Garralón

DIRIGIDO POR: Pedro Pablo Gómez Martín

Trabajo de Fin de Grado

del grado en Ingeniería de Software

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

CURSO ACADÉMICO 2015/2016

UNIVERSIDAD COMPLUTENSE MADRID

AUTORIZACIÓN PARA LA DIFUSIÓN DEL TRABAJO FIN DE GRADO Y SU DEPÓSITO EN EL REPOSITORIO INSTITUCIONAL E-PRINTS COMPLUTENSE

Los abajo firmantes, alumno/s y tutor/es del Trabajo Fin de Grado (TFG) en el Grado en .....de la Facultad de ....., autorizan a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el Trabajo Fin de Grado (TF) cuyos datos se detallan a continuación. Así mismo autorizan a la Universidad Complutense de Madrid a que sea depositado en acceso abierto en el repositorio institucional con el objeto de incrementar la difusión, uso e impacto del TFG en Internet y garantizar su preservación y acceso a largo plazo.

Periodo de embargo (opcional):

- ☐ 6 meses
- ☐ 12meses

TÍTULO del TFG: .....
.....
Curso académico: 20..... / 20.....

Nombre del Alumno/s:
.....
.....

Tutor/es del TFG y departamento al que pertenece:
.....
.....
.....

Firma del alumno/s

Firma del tutor/es

MODELOS DE USO Y RECOMENDACIÓN SOCIAL EN ENTORNOS DE ENSEÑANZA

Resumen

El fin de este trabajo es el estudio y elaboración de un sistema de recomendación basado en el comportamiento de usuarios, concretamente para la página web Acepta el reto, capaz de adaptarse a las capacidades de cada usuario. Esto comprende la creación de un algoritmo para la recomendación de problemas y su implantación en un servicio web.

En el tiempo de desarrollo se ha estudiado la red formada por las relaciones entre usuarios, en base a sus problemas con veredicto correcto. Este trabajo ha servido para definir las bases del algoritmo de recomendación planteado.

Para el despliegue del algoritmo ha sido creado también un servicio web, usando tecnologías actuales, para demostrar su funcionalidad y posible exportación a la página web Acepta el reto.

Palabras clave

Algoritmo, recomendación, red social, servicio web y veredicto.

USAGE MODELS AND SOCIAL RECOMMENDATION IN LEARNING ENVIROMENTS

Abstract

The purpose of this work is the study and deployment of a recommendation system based in user´s behavior, specifically for *Acepta el reto* web page, able to adapt to the capabilities of each user. This comprises the creation of an algorithm for problem recommendation and the implementation in a web service.

In development time it was studied the relationship user´s network, based on his problems with correct veredict. This work has served to define the proposed recommendation algorithm basis

For algorithm deployment has been also created a web service, using current technologies, to demonstrate their functionality and possible export to *Acepta el reto* web page.

Key words

Algorithm, recommendation, social network, web service y veredict.

ÍNDICE DE CONTENIDO

TABLA DE ILUSTRACIONES

1 Introducción

Este trabajo trata sobre la creación de un modelo de recomendación social en un entorno web para la enseñanza, en concreto para la página web Acepta el reto. El fin de este trabajo es la elaboración de un algoritmo de recomendación de ejercicios basado en el comportamiento de los usuarios y su implantación en un servicio web que pueda ser usado por la página web en cuestión.

1.1 Motivación

Se ha elegido este tema por el problema existente, en quienes deciden resolver ejercicios de forma voluntaria para poner en práctica sus habilidades, concretamente a través de un juez online: un sistema en línea para testear código de programación, que se usa también para la práctica de ejercicios de programación, ya que emite veredictos sobre las soluciones enviadas.

Al existir un gran número de posibles problemas que resolver, es fácil para quien decide practicar ejercicios, caer en la indecisión o en una mala elección, como un problema demasiado difícil para sus capacidades o carente de interés y termine por abandonar.

En concreto, este caso puede ser encontrado en la página web Acepta el reto, que alberga una extensa colección de problemas de programación y un juez online que dicta veredictos sobre la colección de las soluciones que le son enviadas. En dicha página los usuarios pueden elegir resolver cualquier problema, pero esta elección puede dar lugar a casos en los que, un usuario sin demasiada experiencia elige un problema que *a priori* le parece interesante pero de demasiada dificultad para él, abandonando la página y posiblemente nuevos intentos de practicar ejercicios de forma voluntaria. Incluso podría darse el caso contrario, de un usuario con mucha experiencia que elija resolver un problema extremadamente fácil, con similar resultado.

1.2 Objetivos

Los objetivos de este trabajo son:

- Buscar en páginas que contengan un juez online y una colección de ejercicios, si existe algún servicio de recomendación de problemas que tomar como referencia.
- Usar métodos de análisis de redes sociales para averiguar la manera de recomendar problemas a los usuarios.

- Crear un algoritmo de recomendación de problemas para los usuarios de la página Acepta el reto, únicamente con la información proporcionada por su base de datos.
- Validar el funcionamiento de dicho algoritmo.
- Implementar dicho algoritmo en un servicio web para poder probarlo y mostrar que posteriormente podrá añadirse a la página de Acepta el reto.

1.3 Organización del trabajo

Inicialmente el equipo para realizar el presente trabajo era de dos personas, pero uno de los integrantes abandonó el proyecto antes de empezar con su parte de desarrollo del mismo, por lo que en la práctica el proyecto ha sido realizado por un único integrante.

Antes del abandono anteriormente comentado, se definió la coordinación que tendría el equipo, optando por una metodología de tipo Scrum ya que se preveían muchos cambios respecto a la idea original y se podía adaptar bien a este proyecto. Esta metodología ágil se tomó por tener entregas parciales con plazos regulares (definidos en dos semanas) que dieran una visión del desarrollo y pudieran servir para mejorar aspectos respecto a la entrega final, además de que continuamente habría que modificar el algoritmo para mejorar su eficiencia. Para la comunicación entre los que iban a ser los integrantes, se adoptó un sistema de mensajería online, aparte de las reuniones antes programadas.

Sin embargo, ante el abandono del miembro antes nombrado, la metodología anteriormente definida se volvió inútil, aunque se adaptó realizando revisiones regulares también cada dos semana, de todo el trabajo hecho hasta ese momento, para al mismo tiempo ir confeccionando lo que sería la memoria.

Para el desarrollo se habilitó un repositorio en Google Drive, compuesto de carpetas de documentación, desarrollo del código y otra con diversos elementos que pudieran ser útiles. Se adoptó para la organización de las versiones de un sistema basado en la fecha de modificación.

Para la elaboración del trabajo y la redacción de la memoria, se ha utilizado documentación principalmente encontrada en Internet, tanto documentos explicativos sobre las tecnologías usadas como documentación técnica de las herramientas necesarias para hacer el presente trabajo.

1.4 Tecnologías usadas

La máquina utilizada para el desarrollo fue un PC, con un procesador Intel i5 a 2,8GHz, con 12GB de memoria RAM y usando el sistema operativo Windows 7 como plataforma para el desarrollo.

El lenguaje para la escritura de código fue Java en su versión 1.8, usando los IDE de Eclipse y NetBeans, además de las librerías proporcionadas por defecto, se usaron las librerías Jersey para en el desarrollo del servicio web y librerías las MySql para la conexión con la base de datos. El sistema de gestión de base de datos usado fue MySql y la aplicación para administrar la base de datos fue MySql Workbench. El proyecto del servicio web se hizo sobre Maven, usando para su testeo el navegador web Firefox.

1.5 Organización de la Memoria

La actual memoria pretende ser una explicación del trabajo realizado, exponiendo los pasos seguidos desde su comienzo hasta su finalización, comentando todo el proceso de elaboración.

Cuenta con el capítulo de Estado del arte, donde se explican los conocimientos usados en la creación del trabajo y que sirven para una mejor comprensión del mismo.

A continuación se describe la página Acepta el reto, de donde partirá este trabajo y que es la base del mismo.

El capítulo siguiente es la explicación del sistema de recomendación de problemas creado. Incluye su elaboración, con un estudio de la red social creada, realizado para determinar la estructura del algoritmo que hará las recomendación, la propia creación del algoritmo y la del servicio web que lo alberga.

El siguiente capítulo detalla las pruebas y validación del algoritmo de recomendación creado.

Por último se exponen las conclusiones a las que se ha llegado y las líneas futuras del presente trabajo.

2 Estado del arte

En este capítulo se expone la fundamentación teórica usada para la elaboración de este trabajo y explicaciones para una mayor comprensión del mismo. Comenzará por la base, que es la búsqueda de sistemas de recomendación de problemas en otros jueces online y la explicación de qué es un sistema de recomendación. Después estará la sección referida a la red social y terminará con explicaciones sobre qué es un servicio web y que conceptos incluye.

2.1 Jueces online

Un juez online es un corrector automático de ejercicios de programación, vía web, capaz de emitir veredictos en base a la solución que se le envía y utilizado en la práctica de ejercicios y en concursos de programación.

Para la elaboración de este trabajo, además de observar el funcionamiento de la página de Acepta el reto, se han investigado otras páginas de resolución de problemas de programación vía web y buscado en ellas un servicio de recomendación de problemas como el requerido para este trabajo, pero no se ha encontrado.

2.1.1 Juez online de la Universidad de Valladolid

Alojado en la página *www.uva.onlinejudge.org/* se encuentra una web preparada para la resolución y corrección automática de problemas (figura 2.1), con ejercicios disponibles para los usuarios y un juez online que evalúa los envíos que hacen dichos usuarios, además de estadísticas de los envíos realizados y rankings de usuarios.

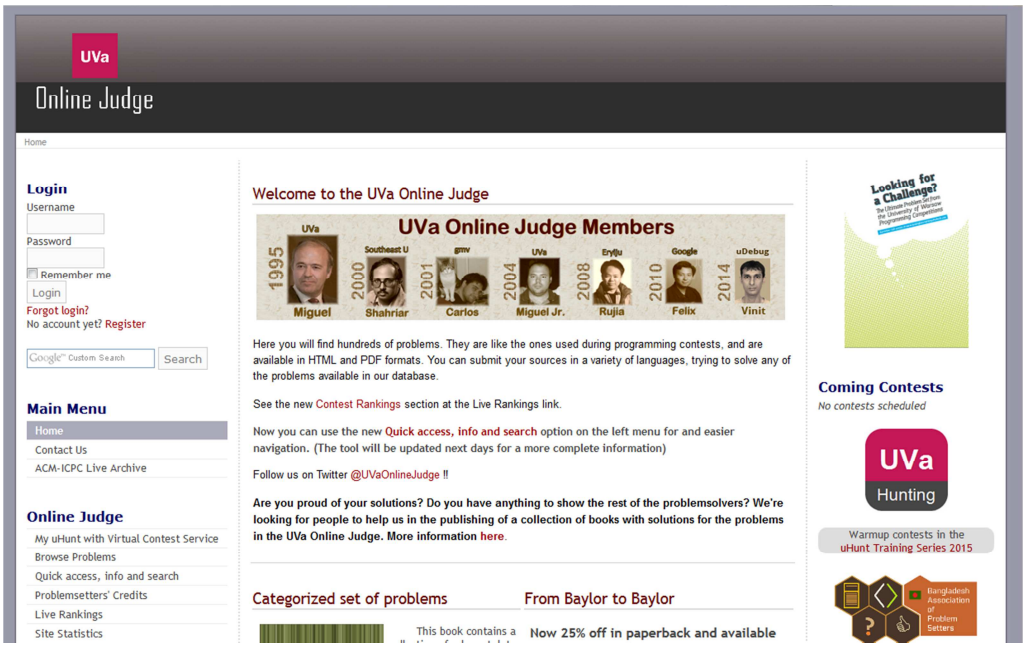

Figura 2.1: juez online www.uva.onlinejudge.org

Dicha página tiene un enlace a “uHunt”, alojado en *www.uhunt.felix-halim.net/,* que proporciona un completo sistema de estadísticas recopiladas a partir de todos los envíos realizados al juez online de la UVa (Universidad de Valladolid), además de estadísticas de usuario (figura 2.2). Incluye varias secciones de interés, como las relacionadas con competiciones de resolución de problemas, aunque la más importante para este trabajo es la de recomendación de problemas. El sistema de recomendación de problemas, basa su funcionamiento en listar los problemas no resueltos por el usuario que hace la consulta, ordenados por el número de usuarios que sí los han resuelto. Es un sistema mínimamente adaptado al usuario y no válido para este trabajo, ya que no recomienda problemas adaptados en las capacidades de cada usuario ni en base a su comportamiento.

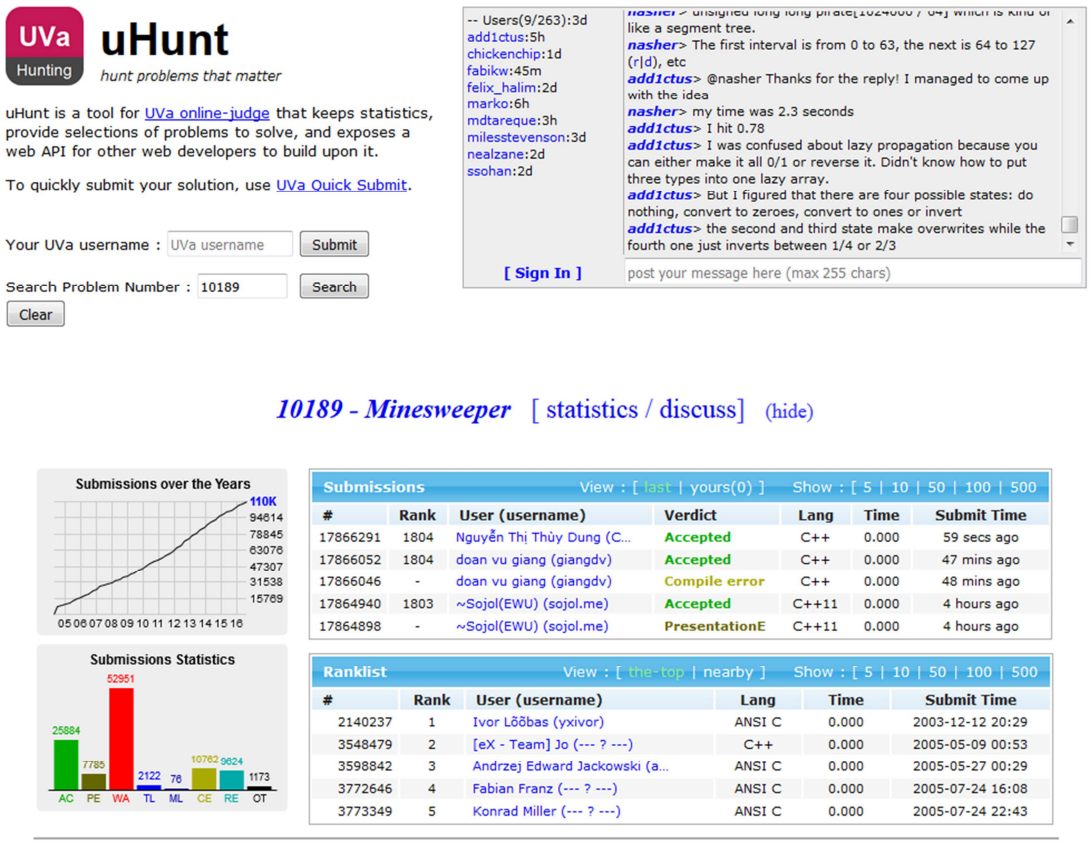

Figura 2.2: juez online www.uhunt.felix-halim.net

2.1.2 Sphere Online Judge

Juez online cuya ruta es *www.spoj.com*, es otro juez online (figura 2.3) entre cuyas características se ha encontrado un indicador de dificultad para cada problema, aunque no un recomendador de problemas en sí mismo.

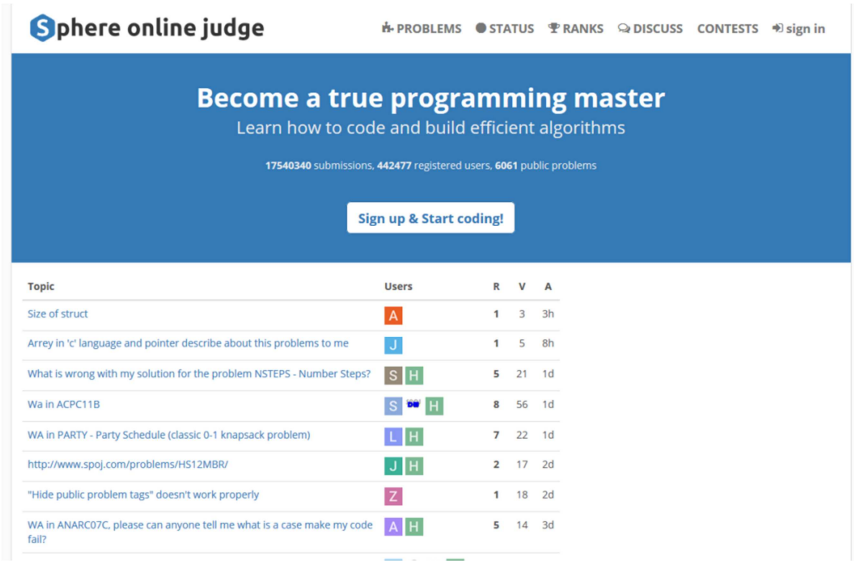

Figura 2.3: juez online www.spoj.com

2.1.3 Uri Online Judge

Juez online disponible en *www.urionlinejudge.com.br*, de apariencia diferente a los anteriores (figura 2.4), con funciones para su uso tanto en entornos académicos como puramente recreativos. Tampoco incluye una sección dedicada a recomendar problemas a los usuarios.

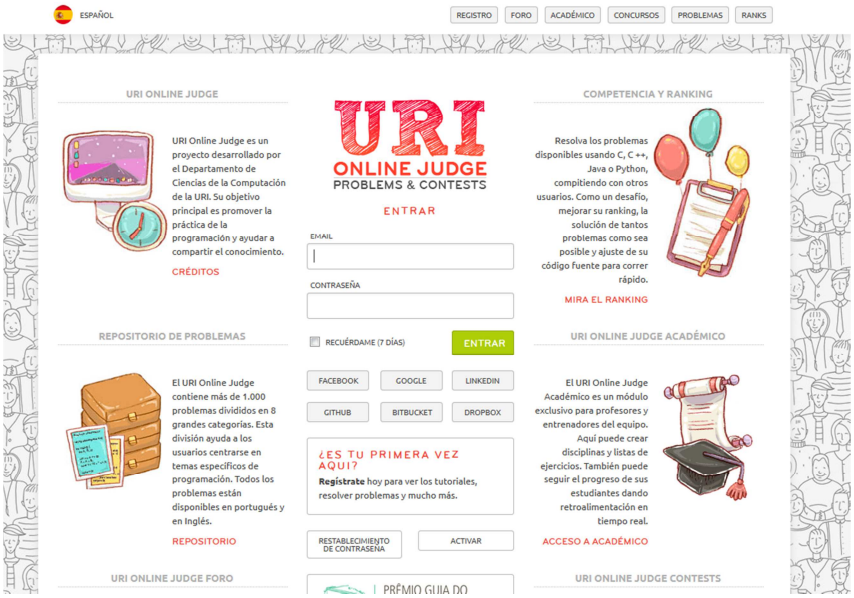

Figura 2.4: juez online www.urionlinejudge.com.br

2.1.4 *Jutge.org*

Página enfocada al aprendizaje de la programación mediante la resolución de problemas, con exámenes incluidos y con la existencia de instructores para facilitar el aprendizaje, además de un juez online para la calificación de los problemas. Incluye cursos y tiene una funcionalidad claramente académica, además de los roles de alumno e instructor para los usuarios es una opción novedosa respecto a otros jueces, pero no incluye un servicio de recomendación de problemas para los usuarios.

Figura 2.5: juez online www.jutge.org

2.2 Sistemas de recomendación

El fin último de este trabajo es la elaboración de un sistema de recomendación para Acepta el reto y ya que no se ha podido encontrar ninguno que se adapte a las capacidades de los usuarios, será necesario montarlo desde el inicio.

Los sistemas de recomendación forman parte de un sistema de filtrado de información, los cuales presentan distintos tipos de temas o ítems de información (películas, música, libros, noticias, imágenes, páginas web, etc.) que son del interés de un usuario en particular (Es.wikipedia.org, 2016 a).

El sistema de recomendación existe dentro de un entorno formado por usuarios y elementos con los que interactúan esos usuarios. Estos elementos pueden tener una puntuación para el usuario, dada al interactuar con ellos y que puede ser puesta de forma explícita o implícita (calculada por sistema).

La función de los sistemas de recomendación se basa en recopilar datos de los usuarios y usarlos para predecir las puntuaciones de elementos sobre los que no ha actuado el usuario, confeccionando un ranking con estas puntuaciones predichas. Adaptándolo al contexto de un juez online, el sistema trataría de adivinar las puntuaciones subjetivas que daría un usuario a los problemas que no ha hecho.

La predicción que hace el sistema puede ser realizada a partir de datos recopilados de los usuarios de forma explícita o implícita.

- De forma explícita, el sistema solicita al usuario su opinión o se le pide hacer una ponderación acerca de un tema, que no sea exactamente sobre el que se va a hacer la predicción.
- De forma implícita, el sistema consulta estadísticas o datos de las actividades que ha realizado el usuario.

La más interesante, en relación con este proyecto, es la forma implícita de recomendar, usando datos de los usuarios que sirvan como indicador de su comportamiento pero sin pedírselos de forma directa.

Otra clasificación dentro de los sistemas de recomendación, es en relación al tipo de datos usados para realizar la recomendación, ya que se aplica un filtro sobre todos los elementos que maneja la aplicación para seleccionar los que sirven para una recomendación. Hay cuatro tipos de filtrado (figura 2.6), tres diferentes y uno combinado:

- Filtrado basado en contenido: la recomendación se basa en el conocimiento de los elementos que ha puntuado el usuario ya, recomendando otros similares a los mejor puntuados.
- Filtrado demográfico: este tipo hace recomendaciones en función de las características de los usuarios como edad, sexo, situación geográfica, etc.
- Filtrado colaborativo: consiste en buscar usuarios similares al que pide la recomendación y de los usuarios relacionados, extraer los elementos mejor puntuados que no haya puntuado el usuario consultor.

- Filtrado Hibrido: combinación de los tres filtros anteriores, pudiendo añadirse alguna otra técnica.

Figura 2.6: Diagrama de los diferentes filtros para sistemas de recomendación. (Jarroba, 2013 a)

De los diferentes tipos de sistemas de recomendación, este trabajo pretende usar uno basado en el filtrado colaborativo, debido a que en la misma definición de este trabajo ya se indica que se usará un modelo de recomendación social, además de que como se observará en el capítulo 3 (Punto de partida: Acepta el reto) y se explicará en el 4 (Sistema de recomendación de problemas), los datos proporcionados hacen que sea este el tipo de sistema de recomendación más apropiado.

Dentro del sistema de recomendación colaborativo, existe una clasificación más:

- Métodos basados en memoria: emplean métricas de similaridad para determinar el parecido entre una pareja de usuarios. Para ello calculan los elementos que han sido puntuados por ambos usuarios y comparan dichos votos para calcular la similaridad.
- Métodos basados en modelos: utilizan la matriz de puntuaciones para crear un modelo a través del cual establecer el conjunto de usuarios similares al usuario activo.

(Jarroba, 2013 b).

El método usado para este trabajo será basado en memoria. Dentro de este, mencionar que existe un algoritmo llamado técnica de los K-vecinos, ligada a los métodos basados en memoria, que basa los elementos recomendados a un usuario en

los que más les han gustado a otros usuarios con gustos similares, vecinos del primer usuario mencionado.

2.3 Análisis de redes sociales

Para poder elaborar este trabajo, será necesario conocer las relaciones de usuarios y problemas dentro de Acepta el reto, ya que como se ha mencionado en la sección anterior, este trabajo se basará en un sistema de recomendación colaborativo.

El análisis de redes sociales trata del estudio de las redes de cualquier tipo de temática, tanto social, como de transportes, de información o Internet. Su estudio se realiza mediante la teoría de grafos y al igual que estos se componen de nodos unidos entre sí por aristas. En el caso de las redes sociales, estas tienen características tales como que los nodos son personas, grupos o cualquier entidad capaz de procesar información y las aristas son las relaciones entre estas entidades.

En estas redes se estudian características como su consistencia y solidez como red, así como las comunidades formadas por los nodos, los cuales representan individuos o entidades como anteriormente se ha explicado, intentando localizar los grupos (cuyos elementos pueden compartir ciertas características) y los *hubs* o concentradores (como elementos con una importancia mayor a la media). De estas medidas se intentan extraer conclusiones extrapolando las métricas tomadas a la realidad.

Las más interesantes para el presente trabajo son:

- Número de nodos y aristas: para conocer el tamaño de la red que se analiza.
- Grado medio: la media de cuantas aristas están conectadas a cada nodo.
- Grado medio con pesos: igual que el anterior pero teniendo en cuenta el peso de las aristas en el cálculo.
- Densidad del grafo: medida de la relación entre nodos tomada en el rango de 0 a 1.
- HITS: permite valorar y clasificar la importancia de las aristas. Muestra la existencia de *hubs* o concentradores: nodos que concentran relaciones de forma masiva.
- *PageRank*: medida creada por Google para asignar relevancia a las páginas web (midiendo su importancia dentro de la red de Internet). De utilidad similar a HITS.

2.3.1 Gephi

Gephi es una aplicación de código abierto escrita en Java, destinada a la visualización y análisis de todo tipo de redes y sistemas complejos con gráficos dinámicos y jerárquicos. Fue creado por estudiantes de la Universidad de Tecnología de *Compiègne* (Francia) y actualmente continúa en desarrollo, siendo la última versión la 0.9. Por su potencia ha sido usado tanto para fines académicos como periodísticos, además de para cualquier otro ámbito que requiera del análisis de grafos y la representación visual de redes de cualquier tipo (Wikipedia.org, 2016).

La herramienta permite tanto la creación de grafos de forma totalmente manual (añadiendo nodos y aristas), como la importación de datos en ciertos formatos para que pueda crear los elementos de la red. A partir de dicha red se pueden tomar métricas para su análisis, incluyendo las necesarias para usar en este trabajo.

Además de poder recopilar estadísticas de la red tiene variadas funcionalidades, como añadir un *timeline* a un gráfico, añadiendo atributos de tiempo a los nodos del grafo y así poder ver su evolución a través del tiempo. Se incluye también una extensa funcionalidad en el ámbito de la visualización de la red, ofreciendo un gran número de opciones diferentes para clasificar los elementos de la red, tanto en color como en tamaño, complementado por un filtro altamente configurable. Además existen algoritmos para la ordenación de la red, alterando su representación y facilitando el análisis visual de ésta, ademas de dar la posibilidad de configurar parámetros de dichos algoritmos según se busque destacar unos u otros aspectos de la red.

2.4 Servicio Web

Un servicio web (en inglés, *Web Service* o *Web services*) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones (Es.wikipedia.org, 2016 b).

Los servicios web permiten que aplicaciones desarrolladas en diferentes lenguajes de programación puedan comunicarse entre sí, usando el canal de Internet. Esto es conseguido gracias a organismos que establecen estándares, que deben usar estas aplicaciones para comunicarse entre sí, siendo esto la causa de su éxito actual y de su cada vez mayor implantación y uso por parte de los usuarios.

Otra ventaja de los servicios web es la posibilidad de ofrecer funcionalidades de forma rápida a los usuarios, sin muchas más condiciones que el acceso a Internet. Logrando por esta una adopción mayor, en detrimento de las aplicaciones residentes en el ordenador de cada usuario. Además de que las aplicaciones que usan de servicios

web tienen una mayor difusión, debido a cada vez mayor acceso a Internet por parte de la población.

Los estándares actualmente más empleados son:

- XML (*Extensible Markup Language*): Formato estándar para el intercambio de datos (entre otros usos), los cuales son almacenados de forma comprensible.
- JSON (*JavaScript Object Notation*): Formato para el transporte de datos usado como alternativa a XML.
- HTTP (*Hypertext Transfer Protocol*): Protocolo usado para la comunicación en las redes.
- SOAP (*Simple Object Access Protocol*): Protocolo que hace posible el establecimiento de un intercambio de datos basado en XML.
- FTP (*File Transfer Protocol*): Protocolo usado para el intercambio de archivos a través de una red TCP (*Transmission Control Protocol*) sencillo y rápido pero de escasa seguridad.
- WSDL (*Web Services Description Language*): Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web (Es.wikipedia.org, 2016 b).
- REST (*Representational State Transfer*): arquitectura que, haciendo uso del protocolo HTTP, proporciona una API (Interfaz de Programación de Aplicaciones o *Application Programming Interface*) que utiliza cada uno de sus métodos (GET, POST, PUT, DELETE, etc.) para poder realizar diferentes operaciones entre la aplicación que ofrece el servicio web y el cliente.

2.4.1 Java Servlet

El *servlet* es una clase en el lenguaje de programación Java, utilizada para ampliar las capacidades de un servidor (Es.wikipedia.org, 2016 c).

Estos *servlet* solo pueden ser usados en los servidores que soporten su API, siendo el uso de esta independiente de la plataforma usada, siguiendo la misma filosofía que Java. Son capaces de generar páginas web dinámicas según las peticiones que reciban, usando las ventajas y posibilidades que ofrece Java como lenguaje de programación.

2.4.2 REST

REST (*Representational State Transfer*), es un estilo de arquitectura de software dirigido a sistemas hipermedia distribuidos como lo es la WWW, *World Wide Web* o

red informática mundial (Es.wikipedia.org, 2016 d). Es un conjunto de principios para la arquitectura de servicios web. Aunque actualmente el término también es usado para nombrar a una interfaz que use HTTP para la comunicación, sin tener capas adicionales como SOAP.

Estos sistemas basados en REST, denominados RESTful, tienen que cumplir sus principios de arquitectura

- Mensajes cliente-servidor sin estado: cliente y servidor están claramente separados y los mensajes que se envían mutuamente, encapsulan toda la información necesaria para que se entiendan mutuamente. Esto evita guardar el estado de la comunicación en elementos como caché o cookies. Además de que aumenta la independencia entre cliente y servidor, que no tienen que almacenar información intermedia.
- Operaciones para el manejo de datos: deben estar implementadas las operaciones CRUD (*Create, Read, Update, Delete*), aunque en el caso de HTTP ya las tiene definidas.
- Interfaz Universal: los recursos son identificados de forma única, a través de su URI (*uniform resource identifier*).

3 Punto de partida: Acepta el reto

A continuación se hará una presentación de la página para la cual se hará el sistema de recomendación. Este juez online da la posibilidad de practicar con gran variedad de problemas y almacena datos de los envíos realizados por los usuarios desde su despliegue, que pueden ser usados en posteriores análisis.

3.1 Página web

La página web Acepta el reto existe en la dirección *www.aceptaelreto.com* y fue creada como iniciativa de varios profesores de la Facultad de Informática de la Universidad Complutense de Madrid y puesta en marcha en febrero del 2014, con la intención de servir como apoyo a estudiantes de programación de cualquier institución.

Figura 3.1: Página principal de Acepta el reto.

La página (figura 3.1) proporciona una serie de problemas de programación de distinta temática, propone su resolución usando diferentes lenguajes e incluye un juez para determinar la validez de dichas resoluciones.

Incluye varias secciones dentro de la misma página como son:

Resolución de problemas: para cada problema se muestra un enunciado explicativo (figura 3.2), además del formato de entrada y salida que deberá tener el ejercicio enviado. Están clasificados por número o por categorías (como se muestra en la figura 3.3), se muestran todos los problemas existentes en la página disponibles para que los usuarios puedan resolverlos. Además se muestran estadísticas de los envíos para cada problema, los autores y la posibilidad de enviar una nueva solución. También se propone en la página principal (figura 3.1) la resolución del “Problema de la semana”, que es un problema seleccionado de entre todos los existentes, para que los usuarios compitan en su resolución. Además de que se muestran disponibles las estadísticas de dicho problema, los autores y la posibilidad de enviarlo para su corrección.

Figura 3.2: Ejemplo de enunciado de problema de Acepta el reto.

Figura 3.3: Sección de problemas clasificados por categorías de Acepta el reto.

Estadísticas: Se incluye también una sección de estadísticas, que registra los envíos realizados por los usuarios (figura 3.4).

Figura 3.4: Lista de los últimos envíos recibidos de Acepta el reto.

Documentación: como es lógico se incluye un apartado para la documentación, que incluye preguntas más frecuentes (figura 3.5), significado de los veredictos, historia de la página de Acepta el reto y mención a los creadores de la página.

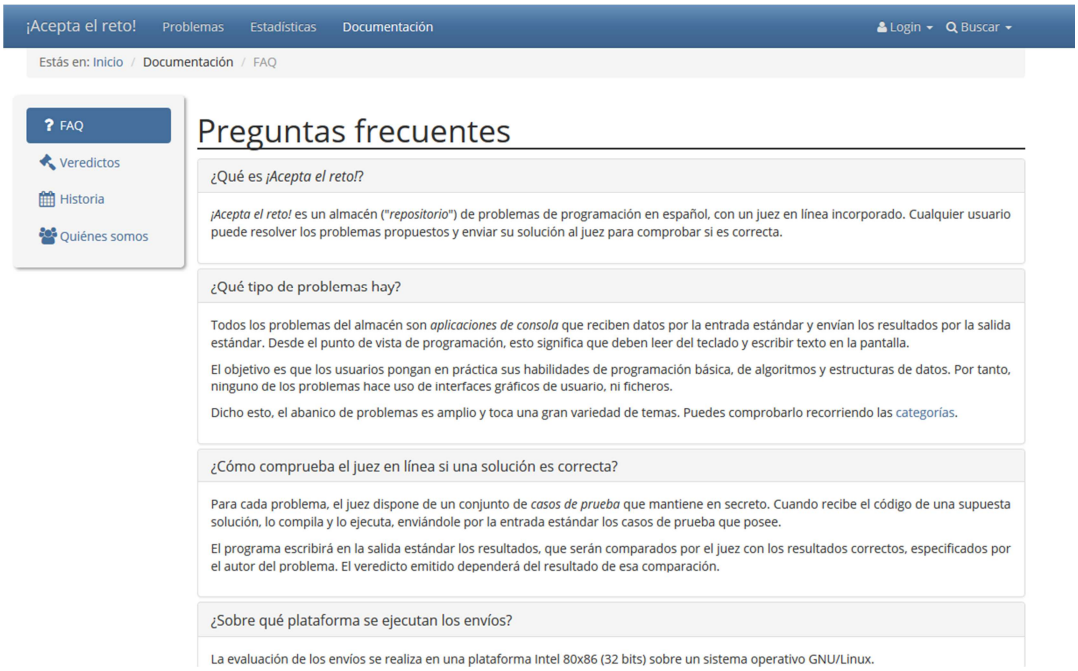

Figura 3.5: Sección de preguntas frecuentes en documentación de Acepta el reto

Elementos comunes: se incluye un buscador de problemas por su identificador y de usuarios también por su identificador. También hay una función de registro y *login* en la página.

3.2 Base de datos de Acepta el reto

De la base de datos completa de Acepta el reto, solo se ha descargado un extracto de esta con todos los campos necesarios para realizar este trabajo (figura 3.6). Por motivos de protección de datos, en el extracto de la base de datos usado, no se incluyen nombres de los usuarios de Acepta el reto. Tampoco se incluyeron los nombres de los problemas de Acepta el reto.

Para comprender los campos de esta base de datos, hay que conocer los veredictos dictados por el juez online:

- AC: acertado.
- PE: error de presentación.
- WA: respuesta incorrecta.
- CE: error de compilación.

- RTE: error en tiempo de ejecución.
- TLE: tiempo límite superado.
- MLE: límite de memoria superado.
- OLE: límite de salida superado.
- RF: función restringida.
- IQ: en cola. Este es temporal, así que no queda registrado.
- IR o IE: Error interno.

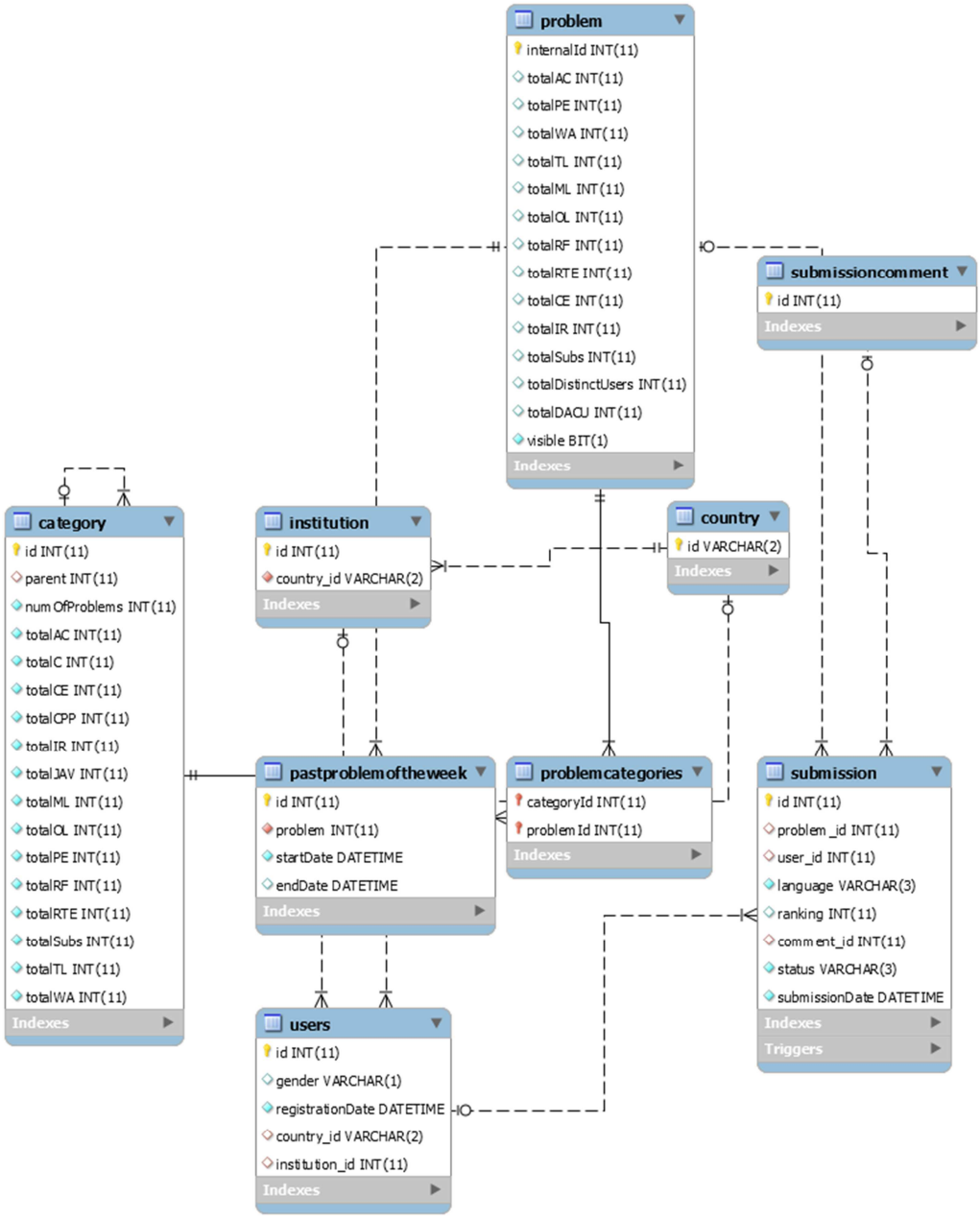

Figura 3.6: Esquema del extracto usado de la base de datos de Acepta el reto.

Como más adelante se explicará, la tabla más importante de todas es *submission*, además del campo *totalDACU* de *problem*, aunque en el primer contacto con la base de datos se consideraron las siguientes tablas para ser usadas en el trabajo:

- category: categorías de los problemas de Acepta el reto con estadísticas de los envíos de problemas de cada categoría.
    - id: identificado de la categoría
    - parent: categoría padre.
    - numOfProblems: número de problemas de dicha categoría.
    - totalSubs: número total de envíos.
    - Resto de campos “total”: total de envíos realizados de los problemas de esas categorías según los veredictos o el lenguaje que indican.
- problem: todos los problemas existentes en Acepta el reto con estadísticas acerca de los envíos realizados de dichos problemas.
    - internalId: identificador del problema.
    - totalSubs: número de envíos totales.
    - totalDistinctUsers: número de envíos de distintos usuarios.
    - totalDACU: número de envíos AC de distintos usuarios.
    - visible: es visible.
    - Resto de campos “total”: total de envíos realizados de esos problemas según los veredictos o el lenguaje que indican.
- submission: datos de los envíos de soluciones a Acepta el reto con campos como lenguaje usado en el ejercicio, fecha de envío o veredicto del envío.
    - id: identificador del envío.
    - problem_id: identificador del problema enviado.
    - user_id: identificador que ha hecho el envío.
    - language: lenguaje en el que se ha hecho el envío.
    - ranking: posición en el ranking.
    - comment_id: identificador de comentario (si es que tiene).
    - status: veredicto de ese envío.
    - submissionDate: fecha del envío.
- users: usuarios registrados en Acepta el reto.
    - id: identificador del usuario.
    - gender: género del usuario.
    - registrationDate: fecha de registro.
    - country_id: identificador del país del usuario.
    - institution_id: identificador de la institución del usuario.

Tal como se muestra en la figura 3.6, existen otras tablas que no son nombradas. Estas tablas no son mencionadas por carecer de información útil para la realización de este trabajo. Algunas tablas se descartaron por motivos obvios, como en el caso de la tabla *submissioncomment*, no es considerada para nada por solo almacenar los comentarios de los usuarios de algunos de los envíos. La tabla *pastproblemoftheweek* tampoco es considerada al incluir información de la sección “Problema de la semana” de Acepta el reto, la cual no tiene relación con este trabajo. Para la tabla *problemcategories*, es por ser claramente una tabla intermedia en la relación entre *problem* y *categories*. El descarte del resto de tablas se explica en el siguiente capítulo.

4 Sistema de recomendación de problemas

En este capítulo se explicará el proceso usado para crear el sistema de recomendación de problemas. Se hará una aproximación a lo que será la base de este sistema, explicando cómo a partir de las relaciones entre usuarios se podrán hacer recomendaciones. Después se hará en estudio de la red social creada a partir de las relaciones entre usuarios, con la idea de definir criterios a la hora de recomendar. Por último se explicarán los pasos seguidos para la creación del núcleo del sistema de recomendación y posteriormente la creación del servicio web que lo albergará.

4.1 Algoritmo de recomendación

El núcleo del sistema deberá ser un algoritmo, que a partir de la información de la base de datos, pueda recomendar problemas a los usuarios atendiendo a ciertos criterios. Éste será nombrado como algoritmo de recomendación y será el encargado de recomendar problemas a los usuarios.

La base para el algoritmo de recomendación buscado, es que a partir de los problemas resueltos por un usuario, se establezcan relaciones con otros usuarios, siendo los problemas que tengan en común lo que los relacionen. Estas relaciones serán tomadas como el indicador de que estos usuarios tienen un comportamiento similar y por tanto, tienen similares gustos y capacidades

Inicialmente se barajó otra alternativa: relacionar problemas a partir de los usuarios que los han resuelto. Pero se descartó al considerar que relacionar usuarios a partir de los problemas que han solucionado es más natural y fácil de comprender y se estimó que el resultado sería similar.

El primer criterio para recomendar problemas al usuario que hará una consulta al sistema, es que los usuarios relacionados con él tengan más problemas resueltos, para así poder recomendar problemas nuevos, que el usuario que ha hecho la consulta no haya hecho.

Para encontrar otros criterios útiles en una recomendación, se decidió hacer un estudio de la red social formada por todos los usuarios, relacionados mediante los problemas que han realizado. Su función sería localizar que factores pueden indicar que dos usuarios tengan el mismo comportamiento y establecer límites para decidir qué relaciones entre usuarios no demuestren que ambos tengan el mismo comportamiento y por tanto, no sean válidas.

4.2 Red social de Acepta el reto

Para estudiar qué criterios usará el algoritmo de recomendación a la hora de relacionar usuarios, con aquel que hace la consulta y posteriormente recomendarle problemas a este, se decidió crear un grafo dirigido y ponderado, en base a las siguientes decisiones.

Dentro de las decisiones tomadas para lograr crear la red social, el primer paso fue descargar la base de datos de Acepta el reto y estudiar las relaciones entre las distintas entidades de la base de datos, especialmente usuarios y problemas ya que eran esenciales para la elaboración del algoritmo. Los datos más útiles para la creación de la red, fueron localizados en las tablas *problem*, *submission* y *category*.

Ya que las relaciones serían entre usuarios a través de los problemas que hubieran realizado, la tabla principal a utilizar fue *submission*, que es la que tiene los registros de los problemas enviados y que relaciona estos con los usuarios y tanto de *problems* como *category* y *users* se podrían conseguir datos adicionales. A partir de esto, se pensaron diferentes tipos de relación entre usuarios:

- Por institución y país de los usuarios.
- Por lenguaje usado en los problemas.
- Por categoría de los problemas enviados.
- Por veredicto de los envíos de problemas.

En primer lugar se descartó usar las relaciones por país e institución de los usuarios, ya que se consideraba totalmente irrelevante cual era el origen de los usuarios respecto a sus habilidades de programación. Esto también dio lugar a dejar de considerar la tabla *users* en el trabajo posterior, ya que únicamente contenía relaciones con la información anteriormente descartada y ningún dato útil adicional.

Poco después también se desechó relacionar por lenguaje, ya que aunque los usuarios dominasen uno u otro lenguaje y con él resolvieran el problema, lo que importa para que un problema sea atractivo a un usuario es su mecánica.

También se dejó de considerar relacionar por categoría del problema, debido a que un problema podía pertenecer a varias categorías. Relacionar usuarios por este factor, daría lugar a un exceso de conexiones, provocando resultados similares para diferentes usuarios, además de no recomendar problemas concretos, sino grupos de problemas. Esta razón llevo a descartar también la posterior utilización de la tabla *categories*.

Por tanto el último factor de recomendación considerado, el de relacionar por veredicto del envío, fue el elegido para crear la red.

Inicialmente ya se pensó que para relacionar dos usuarios con un mismo comportamiento, se podrían relacionar por los problemas que hubieran resuelto de forma correcta. Existió la posibilidad de también considerar los veredictos incorrectos, pero se descartó porque había muchos factores implicados cuando un usuario resolvía erróneamente un problema, prueba de ello son los numerosos veredictos de error, además de que añadiría una complejidad adicional al algoritmo con la contrapartida de un resultado cuestionable.

Teniendo en cuenta lo anterior, las relaciones se crearon entre usuarios que compartiesen problemas con un veredicto “AC” (solución aceptada). Adicionalmente se incluyó para la creación de relaciones el veredicto “PE” (error de presentación), ya que en esencia es el envío de un problema que está bien resuelto, pero que no devuelve el resultado en el formato indicado en el enunciado de dicho ejercicio.

Se encontró el problema de que, cuando un ejercicio ya ha sido enviado y ha recibido un veredicto de resuelto (“AC”), puede volver a enviarse el mismo problema y recibir otra vez un veredicto de resuelto, duplicando el registro de ese problema con ese veredicto (únicamente diferenciados por la fecha del envío). Afortunadamente tenía fácil solución y es seleccionar el primer registro de problema resuelto y descartar el resto.

Se pensó que sería útil incluir como información en la relación entre dos usuarios, cuál de ellos tendría más problemas que el otro, ya que el que tuviera más problemas, podría recomendar ejercicios que no haya hecho el que tiene menos. Esto se pudo hacer gracias a la direccionalidad de las aristas. Se llegó a sopesar la idea de no incluir este tipo de información en el grafo, pero debido a que la capacidad de recomendar es el fin buscado por este estudio, se decidió incluirlo para obtener un análisis más completo. Se decidió también que en el caso de que dos usuarios tuvieran exactamente los mismos problemas, sin tener uno más que otro, no se creara una relación, ya que aunque teóricamente existiera una relación, ambos usuarios no podrían recomendarse problemas entre sí y teniendo en cuenta la naturaleza del sistema de recomendación buscado, este tipo de relaciones no tenía sentido incluirlas.

Se optó también por incluir en los datos usados para crear la red social, la cantidad de problemas hechos que comparten los dos usuarios de una relación, ya que se consideraba una medida de lo similares que son sus comportamientos (a más

problemas resueltos en común, tendrían un comportamiento más parecido) y se tomaría como un indicador de lo sólida o fuerte que sería una relación.

El resultado de estas decisiones fue la creación del grafo dirigido y ponderado nombrado al principio de esta sección, elaborado a partir de todos los usuarios relacionados por los problemas que hubieran hecho en común. Los elementos principales de un grafo son los nodos y las aristas. En este caso concreto, los nodos serían los usuarios y las aristas serían las relaciones entre estos.

Las aristas conectarán usuarios atendiendo a ciertos requisitos:

- Las aristas representarán los problemas en común que tienen ambos usuarios, ya que se considera que para dos usuarios que hayan resuelto los mismos problemas (aquellos con veredicto “AC” o “PE”), hay más posibilidades de que tengan las mismas capacidades, al ser este un indicativo de que tienen el mismo comportamiento.
- Las aristas tendrán una componente de dirección, que indicaría al nodo origen de la dirección, que nodos podrían hacerle una recomendación de problemas (aquellos que son el destino de la dirección de la arista).
- La arista también incluirá peso, calculado a partir de cuantos problemas tendrían en común ambos usuarios.

Por tanto, se decide crear un grafo de la comunidad de usuarios y posteriormente analizarlo, ya que el sistema de recomendación de problemas se basará en encontrar comportamientos similares entre usuarios. Posteriormente, se podrán recomendar problemas en base a esa similitud, siendo esas similitudes relaciones entre usuarios y la mejor manera para analizar esas relaciones es a través de la red social que forman. Las relaciones entre dos usuarios tendrán un factor de desigualdad en el número total de problemas que hubiera hecho cada usuario, ya que esa desigualdad es la que permitirá que dentro de la red, los usuarios puedan recomendarse problemas en un futuro. En el caso de que dos usuarios tuviesen hechos exactamente el mismo número de problemas, no se crearía una relación entre ellos.

El medio usado para el estudio de esta red social fue el programa Gephi, especializado en este fin y que reúne todas las características y funcionalidades para realizar un correcto análisis.

4.3 Creación de la red

Para la creación de la red, primero se extraerá la información ya localizada anteriormente de la base de datos. Después habrá que convertir los datos en forma de nodos y aristas y escribirlos en sendos ficheros, en un formato que se pueda importar a Gephi y por último comprobar la validez del mismo. Todo esto será explicado a continuación.

4.3.1 Formato de los ficheros a importar

Los nodos serán la representación en la red de todos los usuarios que estén registrados y las aristas serán los problemas que tendrán en común los nodos que estas unan. Dichas aristas tendrán una componente de dirección, siendo el origen de la arista el usuario con menos problemas resueltos de los dos y que podría ser recomendado por el otro usuario de la relación. Por otra parte, el usuario al que apunta la arista tendrá más problemas resueltos y podría recomendar los problemas que el otro usuario no hubiera realizado.

El formato de los ficheros a importar será CSV (valores separados por comas o *comma-separated values*), el cual tiene una cabecera que identifica los diferentes datos que habrá y el resto son los propios datos.

El fichero que contendrá los nodos tendrá los campos de:

- Id: identificador del usuario (el mismo que tiene en la base de datos).
- TotalAC: total de envíos correctos.
- PrimerAC: el primer envío correcto.
- UltimoAC: el último envío acertado.

Las aristas por su parte serán de tipo dirigido. Tendrán un nodo que será el origen de la arista y otro nodo que será el objetivo de la arista: el nodo origen es el usuario que menos problemas tiene y podría recibir una recomendación y el nodo objetivo es el usuario con más problemas y que podría recomendar ejercicios nuevos al otro usuario. El fichero de las aristas tendrá los siguientes campos:

- Id: identificador de la arista.
- Source: identificador del nodo que será el origen.
- Target: identificador del nodo que será el objetivo.
- Type: tipo de arista (todas serán de tipo dirigido).
- Weight: peso de la arista. Serán los problemas en común que tendrán los nodos que una.

En un principio se pensó en incluir el campo de fecha de cada envío para poder usar la función de *timeline* que proporciona Gephi y gracias a ella ver la evolución de la red de forma visual. Pero se encontró la imposibilidad de que Gephi separara cada envío en diferentes aristas (sólo creaba aristas con un peso fijo, incapaz de variar el peso de esas aristas, que era lo buscado), por lo que se abandonó la idea de incluir *timeline*.

Pese a no utilizar la función del *timeline*, se pensó que sería útil ver la evolución de la red con el paso del tiempo, así se creó la posibilidad de escribir ficheros de aristas con todos los registros hasta un identificador de envío límite (punto en el *timeline*).

4.3.2 Aplicación para crear los ficheros

Al tener muchos pasos y para simplificar el proceso, se hace una aplicación simple en Java, que se pueda manejar desde la consola y que haga todas las tareas hasta obtener los ficheros de nodos y aristas. La aplicación se llama “AppAuxiliarGephi” y existe un manual para el uso de dicha aplicación en el anexo A.

Esta aplicación Java incluye las siguientes funciones:

- Realizar las conexiones con la base de datos.
- Hacer una consulta que extraiga los usuarios y demás datos de los nodos.
- Hacer las consultas y las operaciones necesarias para obtener los datos de las aristas.
- Escribir los datos en ficheros de formato CSV.

Para el fichero de nodos los datos a aportar son bastante básicos y pueden ser extraídos mediante una única consulta sobre la tabla *submission*.

En cambio para el fichero de aristas, la consulta a realizar es algo más completa. Primero se extraen los usuarios que harán de origen de la arista y con el identificador de cada uno de los usuarios se realizará otra nueva consulta para extraer los usuarios objetivos de la tabla *submission* y con los requerimientos anteriormente citados en la sección 4.2 de este capítulo.

Para la creación de los ficheros de aristas limitados por periodo de tiempo, se añadió la opción de limitar por porcentaje del *timeline* de todos los registros, es decir, desde los registros iniciales hasta el porcentaje indicado de todos los registros existentes.

4.4 Análisis de la red

A continuación se realizará un análisis de los datos obtenidos de la red social de Acepta el reto, explicando cómo fue importada a Gephi y más adelante estudiando

aquellas características consideradas de más interés para este trabajo. La sección terminará con las conclusiones prácticas sacadas del análisis

4.4.1 Importación a Gephi

La aplicación Gephi da la opción de importar ficheros de nodos y aristas de formato CSV (valores separados por comas o *comma-separated values*) y con ciertos campos requeridos, como en el caso de los nodos, el valor de cada nodo y para las aristas el origen y destino de estas.

En el último paso de la importación daba la opción de modificar el formato de los datos de algunas columnas (no todas), así que de las que se pudo, se cambió el formato de texto a numérico, para que fuera más sencilla la consulta de los datos. Tanto en nodos como en aristas se creó un nuevo campo *label* vacío de forma automática, el cual puede visualizarse en la representación de la red, por lo que según conviniera se podrían copiar los datos de otro campo a ese, obteniendo una imagen más rica de la red.

4.4.2 Representación de la red

La red creada a partir de la totalidad de los datos obtenidos de la base de datos, constaba de 1489 nodos y 288280 aristas (figura 4.4).

Para la representación se eligió una distribución Force Atlas 2, un algoritmo de distribución de nodos con parámetros modificables (figura 4.1), elegido porque permitiría ver la formación de comunidades, en el caso de que las hubiera, realiza esa distribución de forma rápida y permite parar su ejecución en el momento que se quiera.

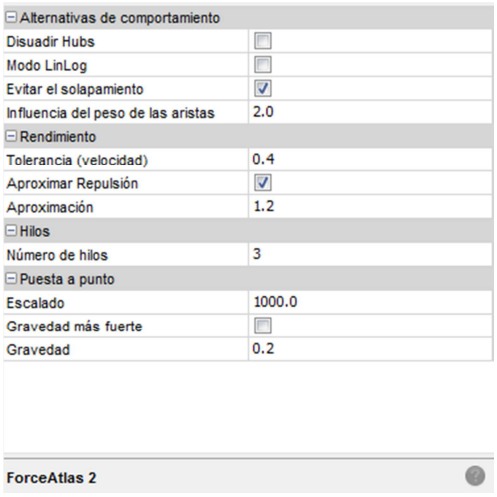

Figura 4.1: Opciones tomadas para la visualización de la red social de Acepta el reto en Gephi.

Después se hizo una clasificación visual de los nodos que pudieran ayudar a sacar conclusiones durante el análisis. Recordando que una arista apunta al nodo que podría recomendar problemas a su origen, se consideró darle importancia a la medida de grado de entrada, ya que representa las veces que un nodo podría recomendar problemas a otros, que además es ponderada con la solidez de esa relación (representada por el peso de la arista), en el grado de entrada con peso, siendo esta última medida la capacidad de recomendar problemas de un nodo con la seguridad de que fueran de su interés. Así que los nodos se clasificaron por tamaños según la medida del grado de entrada con peso (a mayor tamaño, mayor grado de entrada con peso).

También se modificó la apariencia de los nodos, en concreto su color, en base al total de envíos correctos del nodo (campo “TotalAC”), siendo coloreado más oscuro cuantos más problemas hubiera resuelto el usuario representado por el nodo. Fue considerando útil observarlo junto al grado de entrada con peso, ya que inicialmente se supuso que un usuario, a mayor número de ejercicios resueltos, tendría mayor capacidad de recomendación a otros usuarios y se quería comprobar dicha suposición.

Con todo esto se obtuvo la representación observada en la figura 4.2.

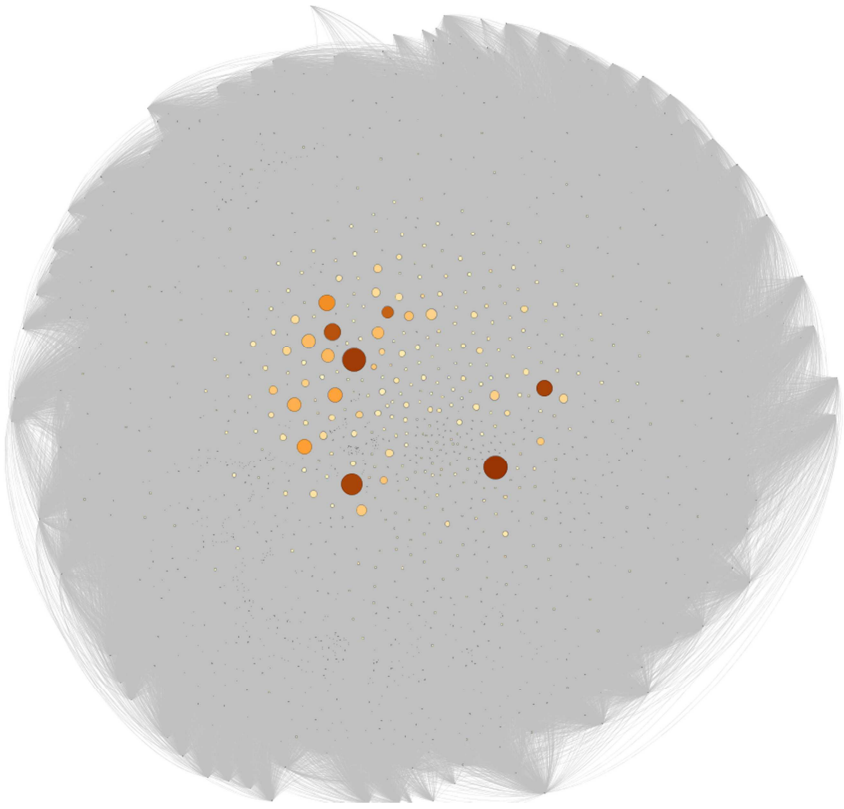

Figura 4.2: Representación inicial de la red social de Acepta el reto en Gephi.

Dado que tanto para los grados con pesos como el total de aciertos, no se ha observado que tengan una distribución lineal, sino que se distribuyen en extremos, se modificó la forma en la que se aplica la clasificación para una mejor visualización. La opción a modificar dentro de Gephi fue el *spline*, para el cual se eligió una gráfica curva en la clasificación por pesos y colores. Los criterios para dar peso y color a los nodos fueron los mismos que para la anterior representación.

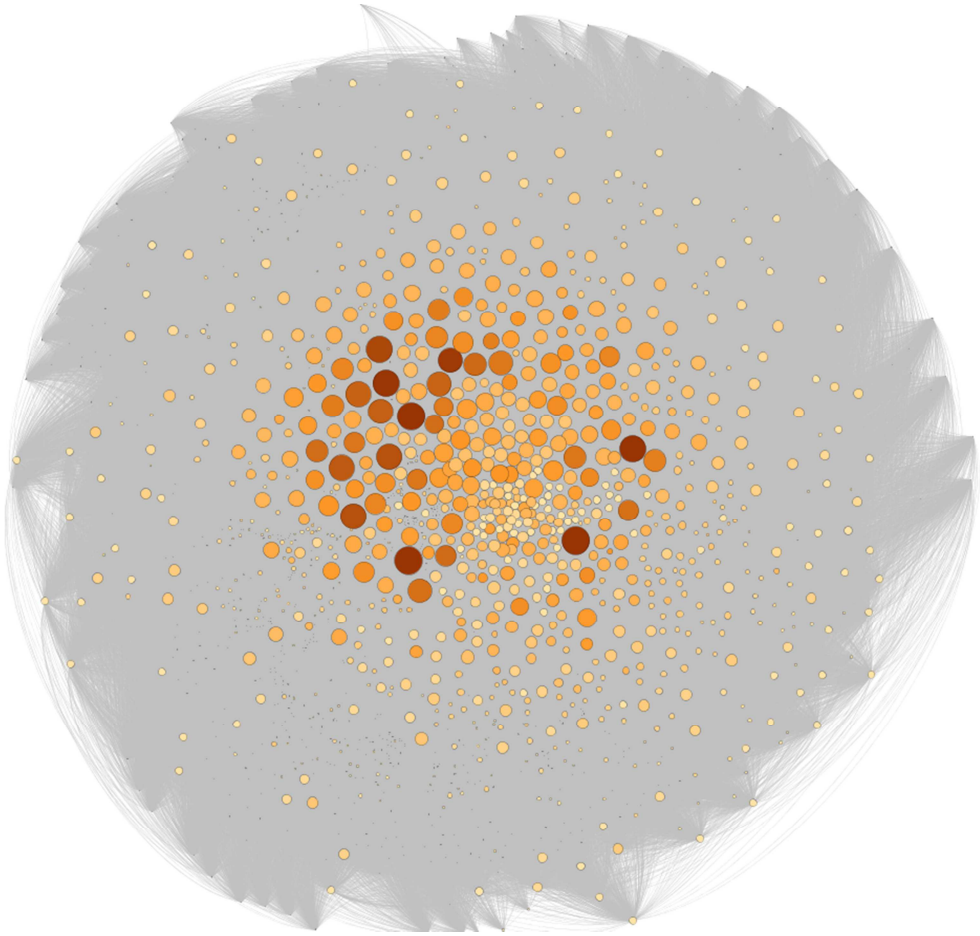

Figura 4.3: Representación de la red social de Acepta el reto en Gephi.

Pese a haber un exceso de aristas, se consideraba que el resultado (figura 4.3) era una representación válida para el análisis. En la observación inicial de la red, se corroboró visualmente la suposición de que a mayor número de problemas resueltos tenga un usuario, tendrá mayor capacidad de recomendar problemas a otros usuarios. Este hallazgo será analizado más profundamente en la sección 4.4.3.2. En el tema de formación de comunidades, no se observó que hubiera agrupaciones claramente definidas.

4.4.3 Medidas obtenidas de la red

Se hizo un cálculo de las medidas de la red a través de Gephi (figura 4.4), con el fin de que ayudara a crear el algoritmo de recomendación.

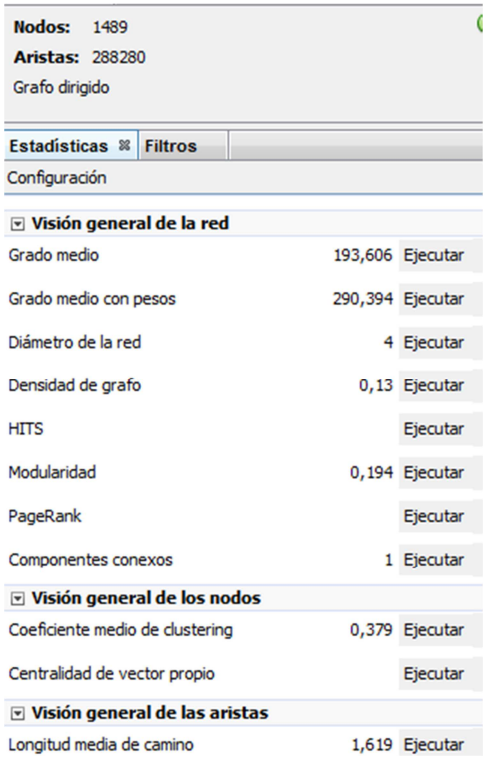

Figura 4.4: Lista de medidas obtenidas de la red social de Acepta el reto.

4.4.3.1 Grado medio y nodos de mayor grado

El valor del grado medio sobre todos los nodos de la red fue de 193,606. Significa que cada usuario se relaciona de media con casi otros 194 usuarios.

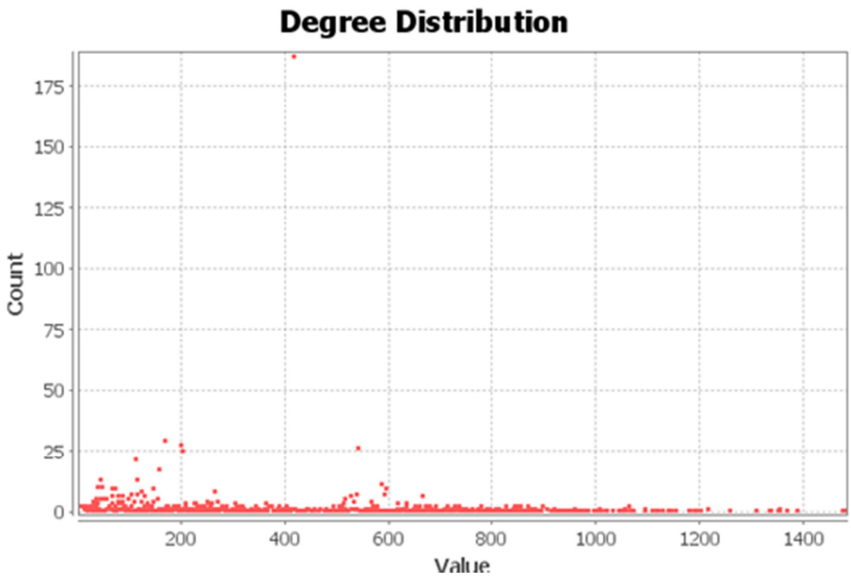

Figura 4.5: Gráfica de la distribución de los grados en la red social de Acepta el reto.

En la gráfica (figura 4.5), el grado de los nodos se encuentra bastante distribuido y no hay muchos nodos que compartan grado, aunque hay un pico entorno al grado 400 que será analizado más adelante. Salvo ese detalle, se esperaba esa variedad de resultados.

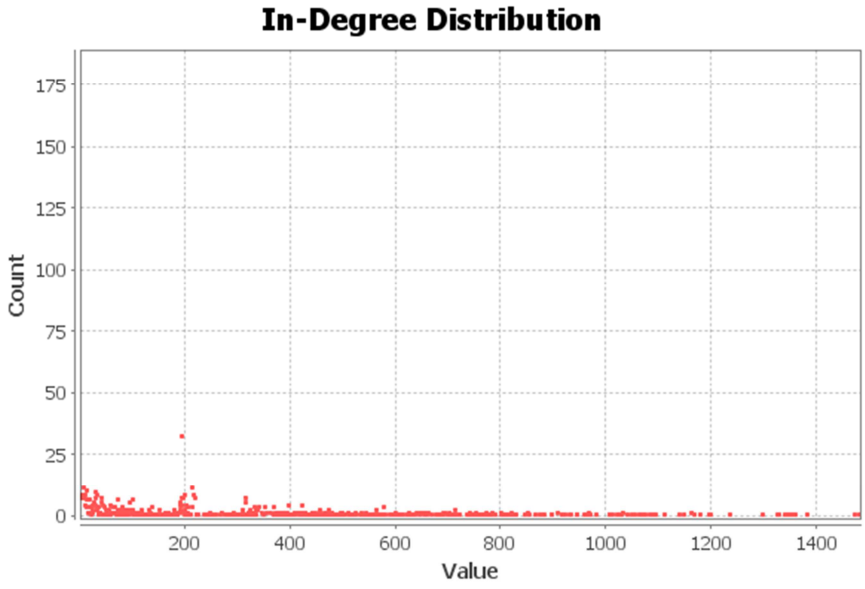

Figura 4.6: Gráfica de la distribución de los grados de entrada en la red social de Acepta el reto.

En la gráfica de distribución de grados de entrada (figura 4.6), siendo grado de entrada los usuarios a los que podría recomendar un nodo, hay una gran variedad de resultados, estando la mayor parte de los nodos por debajo de la mitad de la escala, lo que da a entender que la mayoría de los nodos tienen poca capacidad de recomendación, es decir, tienen menos ejercicios resueltos que la media y menos de la mitad tiene más problemas resueltos que la media.

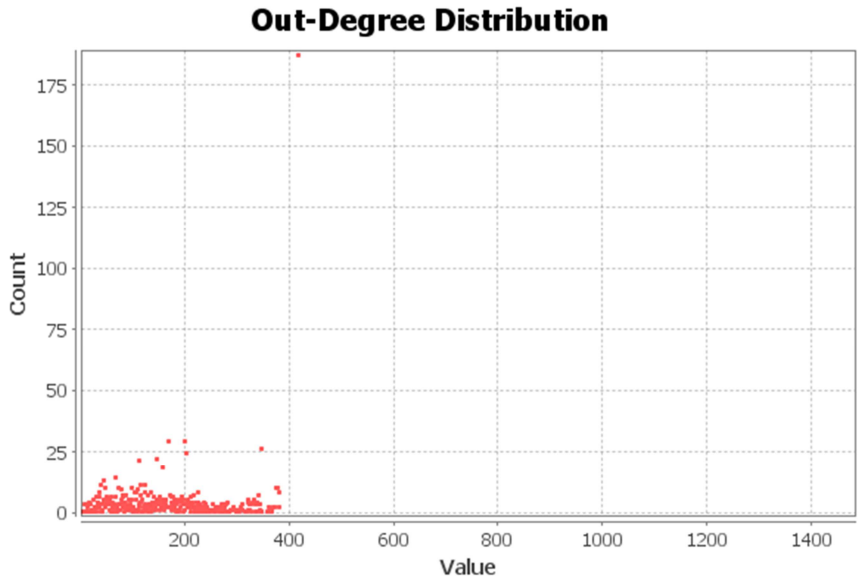

Figura 4.7: Gráfica de la distribución de los grados de salida en la red social de Acepta el reto.

En la distribución de grados de salida (figura 4.7), que son los usuarios que podrían recomendar a un nodo, se vuelve a ver un pico entorno al valor 400 (como en la distribución de grados), estando el resto de los nodos por debajo de ese valor. Para hallar la causa del inusual pico en la gráfica de los grados de los nodos, especialmente en los grados de salida, se han observado directamente los atributos de los nodos.

Y es que aquellos nodos con un único problema resuelto satisfactoriamente, son recomendados de forma masiva. Se han observado series más pequeñas de nodos que comparten el grado de salida y en su mayoría tienen un único envío. También se observó que los nodos con mayor grado se corresponden con aquellos de mayor grado de entrada y con mayor número de envíos satisfactorios. El mayor grado es el del usuario 62, que también es el segundo con más envíos de problemas acertados.

4.4.3.2 Grado medio con pesos

El grado con peso de un nodo se calcula obteniendo las aristas que inciden en él y sumando los pesos de estas aristas.

El valor del grado medio con pesos fue de 290,394. El primer dato que da esta medida es que de media, un nodo tiene relaciones con otros en las que comparte 290 problemas en común de media y de las que seguramente varias tengan la suficiente solidez como para determinar que ambos nodos tienen el mismo comportamiento.

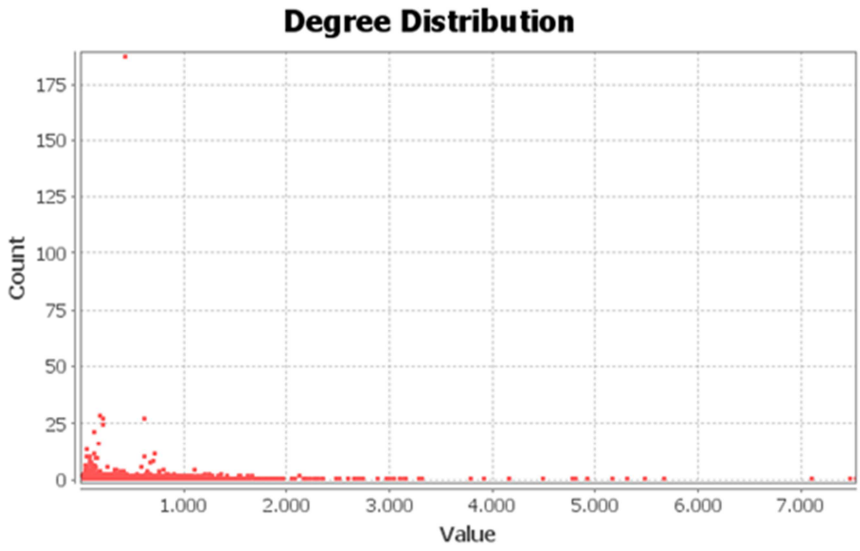

Figura 4.8: Distribución de los grados con pesos en la red social de Acepta el reto.

Esta gráfica (figura 4.8) es muy útil para observar la fuerza de las relaciones entre los nodos de la red. La gráfica podría clasificarse como una mezcla de una campana de Gauss y una cola larga, dejando ver que el grado con peso de la mayor parte de los nodos, pese a ser algo pequeño respecto a la escala, es mayor que 0, por tanto podría ser la prueba de que buena parte de las relaciones tienen una fuerza aceptable, siendo sostenible un sistema de recomendación. Al igual que con los grados sin peso de los nodos, la mayoría se acumulan en valores bajos, mientras que hay unos pocos con valores altos.

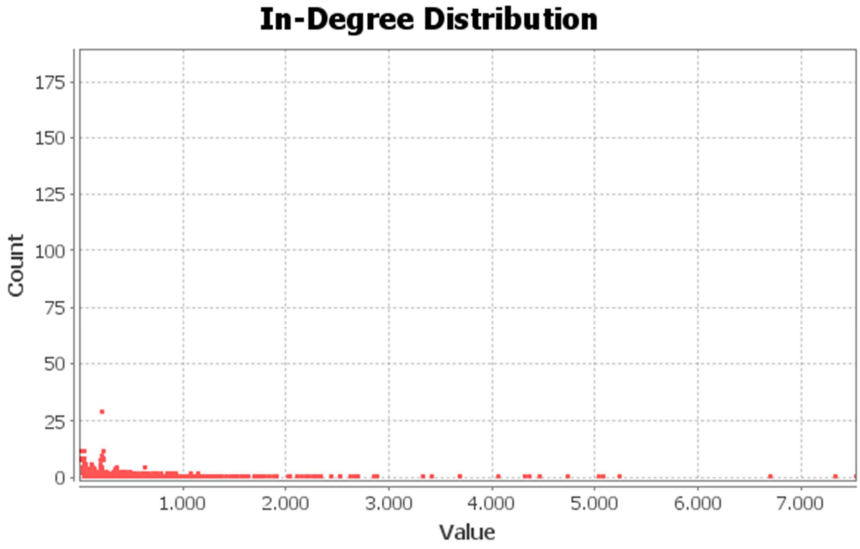

Figura 4.9: Distribución de los grados de entrada con pesos en la red social de Acepta el reto.

La gráfica de los grados de entrada con peso (figura 4.9), al igual que los de sin peso, vuelven a estar muy repartidos, estando la mayoría en la parte baja de la escala y llegando a similares conclusiones.

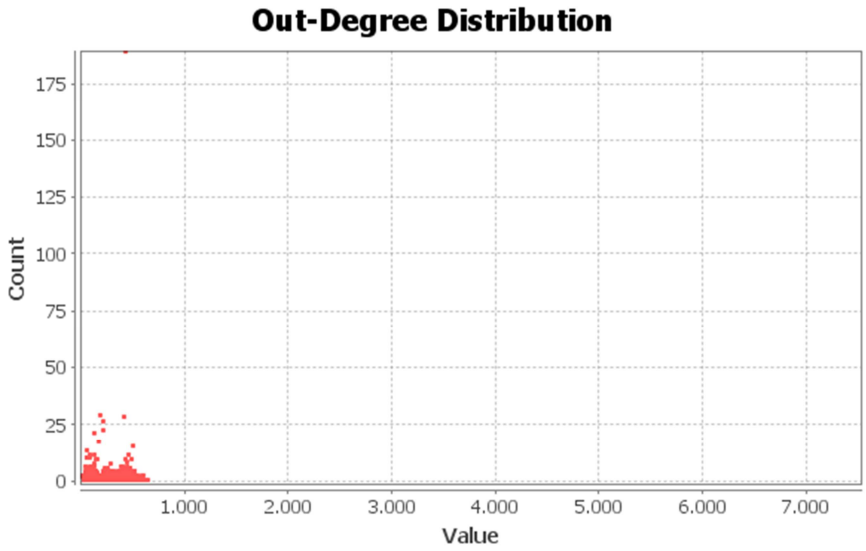

Figura 4.10: Distribución de los grados de salida con pesos en la red social de Acepta el reto.

En la distribución de los grados de salida (figura 4.10), se ve que estos están apelmazados en la parte baja de la escala, no acercándose al grado 1000 y también se observa algún pico que puede que esté relacionado con el observado en el grado de salida sin peso.

Uno de los objetivos es analizar las relaciones de más calidad. Este indicativo de la calidad es el peso de las aristas, por lo que el grado medio con pesos es de gran importancia. Por lo que se analizaron también los datos en bruto.

Aquellos con menor grado de entrada con peso se corresponden con aquellos que tienen menos envíos correctos, en cambio el grado de salida con peso parece que no tiene ninguna relación con los envíos realizados correctamente.

De la misma forma aquellos con mayor grado de entrada con peso están directamente relacionados con aquellos con mayor número de envíos acertados. Al mismo tiempo aquellos con los menores grados de salida sin peso tienen valores muy bajos, en contraste con los grados de salida con peso, que tienen valores bastante altos. Esto puede ser debido a que aquellos usuarios con más problemas realizados, podrían recibir pocas recomendaciones, al no haber muchos usuarios que superen su número

de problemas acertados, pero aquellos que pudieran recomendarles (que tuvieran más problemas resueltos), tendrían muchos problemas en común, habiendo grandes probabilidades de que tengan un comportamiento parecido y aumentando las métricas.

4.4.3.3 Densidad

El valor es de 0,13. Indica que los nodos tienen poca relación con el conjunto, demostrando que los usuarios tienen criterio al establecer conexiones, además de que hay un número muy alto de usuarios como para que estén todos interrelacionados. Es una red dispersa.

4.4.3.4 *Pagerank*

Esta medida sirve para localizar nodos de gran relevancia en la red, es decir, nodos concentradores o *hubs* que tengan un número mucho más alto de conexiones con otros nodos que la media.

La existencia de nodos concentradores puede también descubrirse mediante la medida HITS, pero para esta red concreta, los resultados son más legibles con *pagerank.*

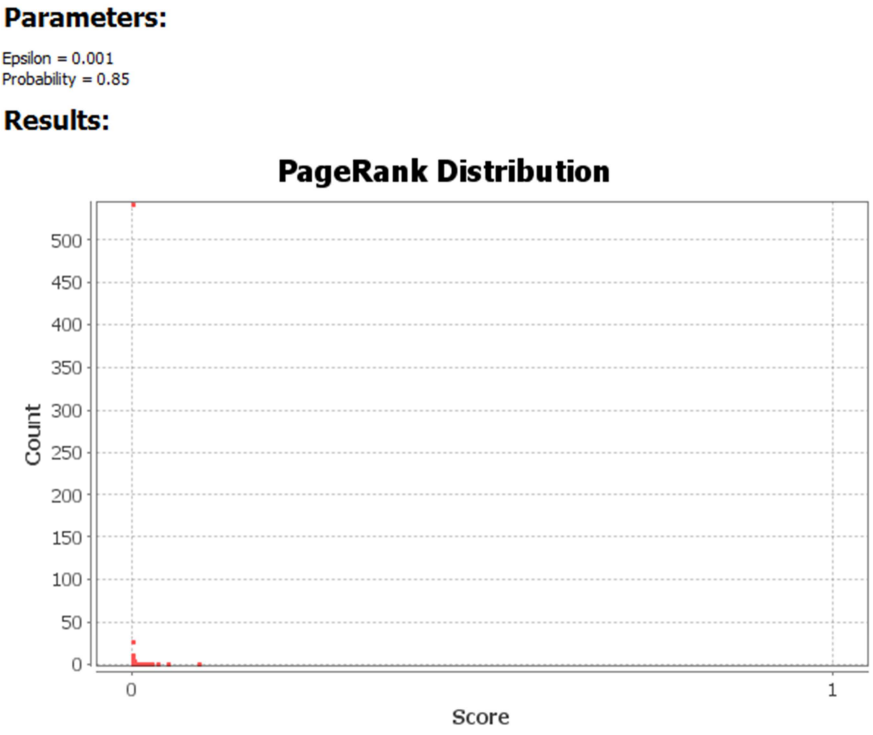

Figura 4.11: Gráfica *pagerank* de la red social de Acepta el reto.

Como se ve en la gráfica 4.11, la mayor parte de los nodos tienen una importancia muy cercana a 0 y solo unos pocos se alejan del origen de coordenadas. Los nodos con más puntuación tampoco destacan de manera excesiva y son aquellos con más envíos de problemas resueltos.

En el contexto del sistema de recomendación, significa que no hay nodos imprescindibles y que su desaparición no debería influir en los resultados de manera excesiva.

4.4.3.5 Pesos de las aristas

El hecho de que más de dos tercios del total de aristas de la red eran de peso 1, sumado a que algo más de la mitad de los usuarios tenía sólo uno o dos envíos válidos de problemas, daba a entender que podía haber un problema respecto a la calidad de las conexiones entre nodos, además de ocasionar una cantidad considerable de ruido en el grafo.

Se pensó que una razón posible de tantas conexiones de peso 1, era la de que la mayor parte de los usuarios compartían el mismo problema, por lo que se consultó directamente el origen de los datos.

El campo buscado era *totalDACU*, nombrado en la sección 3.2 del capítulo Punto de partida: Acepta el reto, que indica el número de usuarios diferentes que han realizado correctamente el problema. Se observó que el problema con mayor valor en este campo es el 39, que corresponde al problema ¡Hola mundo!, considerado como el problema más sencillo de Acepta el reto. Es más, en los registros consultados, tenía casi tres veces más envíos que el siguiente en la ordenación.

También se observó que aquellas aristas de más peso son aquellas que hay entre los nodos de mayor grado sin peso, mayor grado de entrada con peso y mayor número de envíos correctos.

Esto refuerza la idea de que la calidad de las relaciones entre usuarios crece conforme al número de problemas que han enviado y han sido calificados como resueltos.

4.4.4 Acotación de las aristas de la red

Debido al descubrimiento hecho en la anterior sección, de que la mayoría de las aristas de peso 1 no tienen ningún valor, al relacionar a muchos usuarios por el mismo problema, se decidió filtrar las aristas de peso igual o menor a 1 con la intención de detectar cuáles son los envíos totales realizados por los usuarios que se hallen en el extremo de lo que se considera una relación consistente. Además de observar si en la

nueva red filtrada, varía alguna medida de forma significativa, arrojando nuevas conclusiones.

Filtrando los nodos con grado menor o igual a 1, quedó una nueva representación de la red con menos ruido provocado por las aristas (figura 4.12), en la cual casi todas las medidas habían variado pero cuyas gráficas conservaban la misma forma, sólo con la excepción de los grados, cuyas gráficas se han mostrado algo más legibles, aunque arrojando similares resultados.

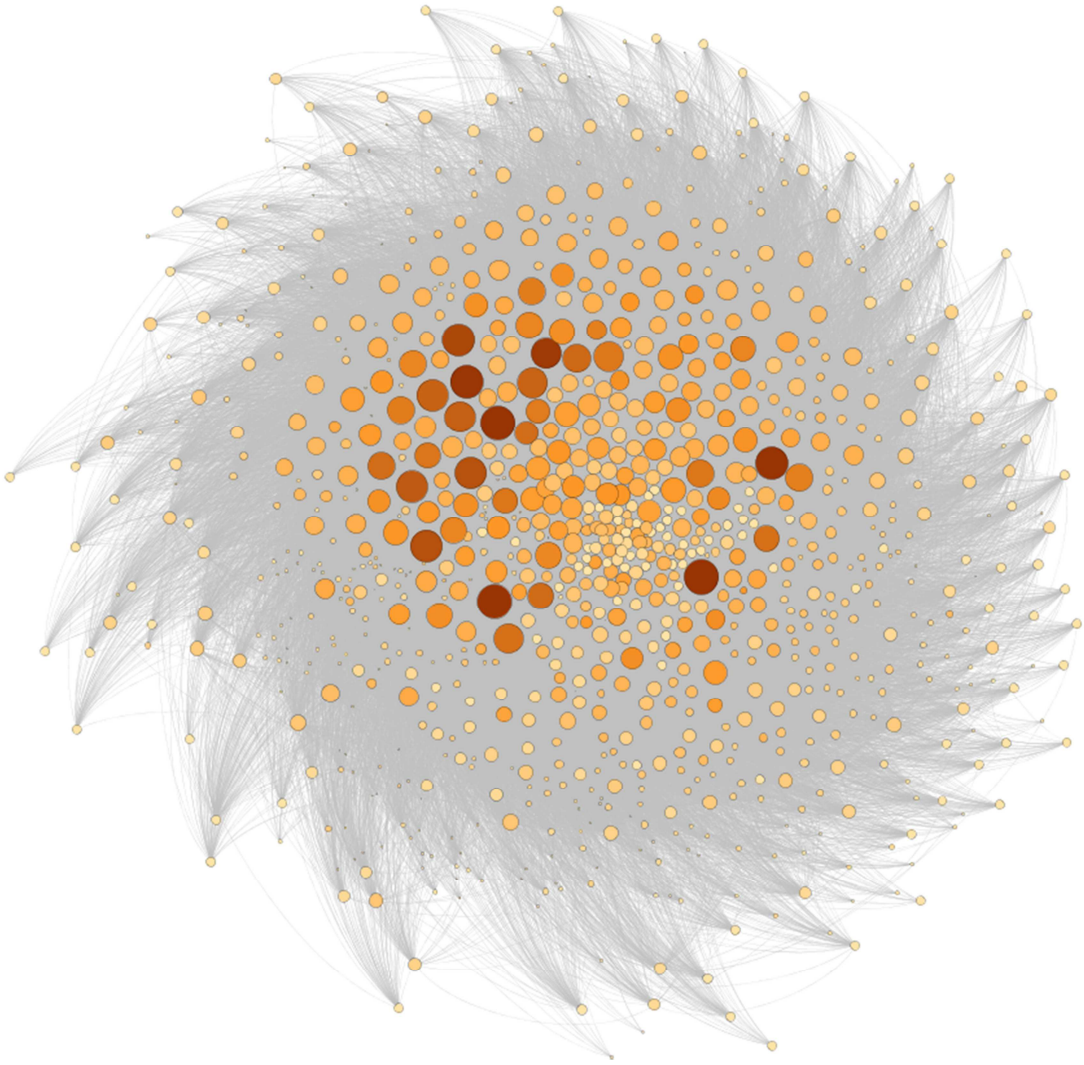

Figura 4.12: Red de Acepta el reto, filtrados nodos menores o iguales a 1.

En esta nueva representación se ha reducido de forma visible la cantidad de aristas, y también en menor medida, la cantidad de nodos existentes (aquellos con grado 1 y 0 han sido excluidos).

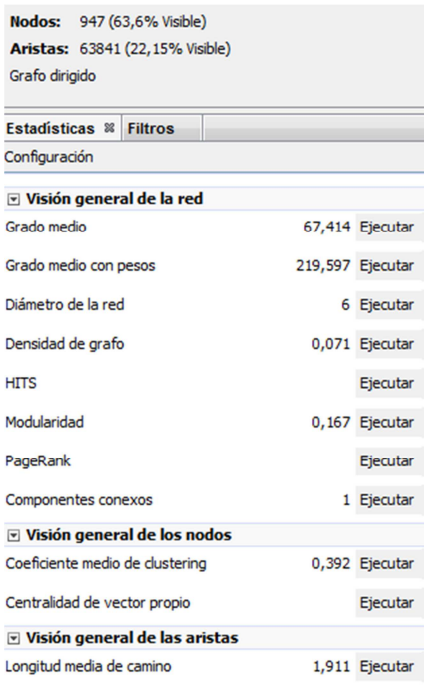

Figura 4.13: Medidas de la red Acepta el reto filtrando nodos menores o iguales a 1.

Tal como se ve en las métricas de la figura 4.13, la densidad de la red ha pasado de 0,13 a 0,071, siendo este un descenso a casi la mitad de la densidad de la primera gráfica y notándose en la propia representación de la gráfica. Esto corrobora que el filtrado de la red ha sido un éxito y se ha limpiado el grafo de conexiones inútiles que solo añadían ruido.

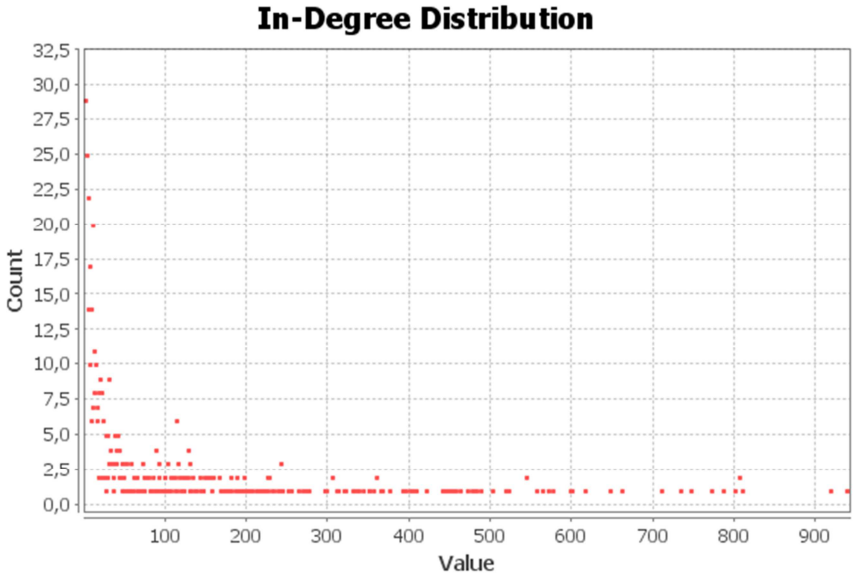

Figura 4.14: Gráfica de la distribución de los grados en la red de Acepta el reto habiendo filtrado los nodos.

La distribución de grados de entrada sin peso (figura 4.14) es la gráfica que ha cambiado más, respecto a la de la red sin filtrar anteriormente analizada. Muestra una forma de cola larga, a diferencia de la anterior, en la que los nodos se repartían más uniformemente por la gráfica, no compartiendo un gran número el mismo grado de entrada. En esta gráfica se puede observar que hay una gran parte de los nodos que podrían hacer pocas recomendaciones de problemas. Viendo estos resultados, se decidió que sería necesario filtrar esos nodos con un grado tan bajo, ya que se les consideraba poco fiables para recomendar.

Se hizo un estudio de los nodos con menores grados de salida y menores grados de entrada, pero ambos por encima de 1, ya que estos eran los usuarios que se hallaban el límite de ser recomendados y recomendar. La mediana de los nodos con menor grado de salida por encima de 1 fue de 2 y la misma medida para los de menor grado de entrada fue de 4, por lo que 4 y 2 podrían considerarse como los problemas resueltos mínimos para poder recomendar y ser recomendados respectivamente.

4.4.5 Evolución de la red a través del tiempo

Se propuso además, con el fin de afinar la mecánica del algoritmo de recomendación, observar la evolución de las aristas de más peso (relaciones más fuertes) a lo largo del tiempo para confirmar o desmentir la suposición de que aquellas aristas de más peso siguen aumentando ese peso a lo largo del tiempo. Significaría que aquellos usuarios con más relación pueden hacer mejores recomendaciones, ya que si a lo largo del tiempo aumentan los problemas que han resuelto ambos, los usuarios entre los que hay una relación mínimamente fuerte van a seguir fortaleciendo esa relación, al coincidir en gustos. En el caso contrario, las aristas de más peso no aumentarían o lo harían de forma muy pobre, entendiendo que las mejores recomendaciones serán las que se hagan en base a todos los usuarios con los que tenga relación, aunque esta sea pequeña, apoyando la premisa de que la mayoría no se equivoca.

Para observar la evolución de la red a través del tiempo, al no poder usarse la función del *timeline*, se obtuvieron los datos para tres nuevas redes. *Timeline* sirve para designar el periodo de tiempo existente desde el primer problema enviado correctamente a Acepta el reto hasta el último problema acertado antes de la descarga de la base de datos, usada para este trabajo. El referenciar un porcentaje del *timeline* en un elemento, significa que ese elemento usa datos desde el inicio del *timeline* hasta el punto correspondiente al porcentaje dicho, siendo descartado el resto. Las redes creadas son correspondientes al 40, 60, 80 y 100 por ciento del *timeline* (la red anteriormente estudiada es la que contiene el 100% del *timeline*).

Para mostrar claramente los pesos de las aristas, que es el dato que más interesa observar, se filtró por peso de la arista mayor que 30, lo que dejó un número suficientemente reducido de aristas para facilitar la observación de cambios, y por grado igual o mayor a 1.

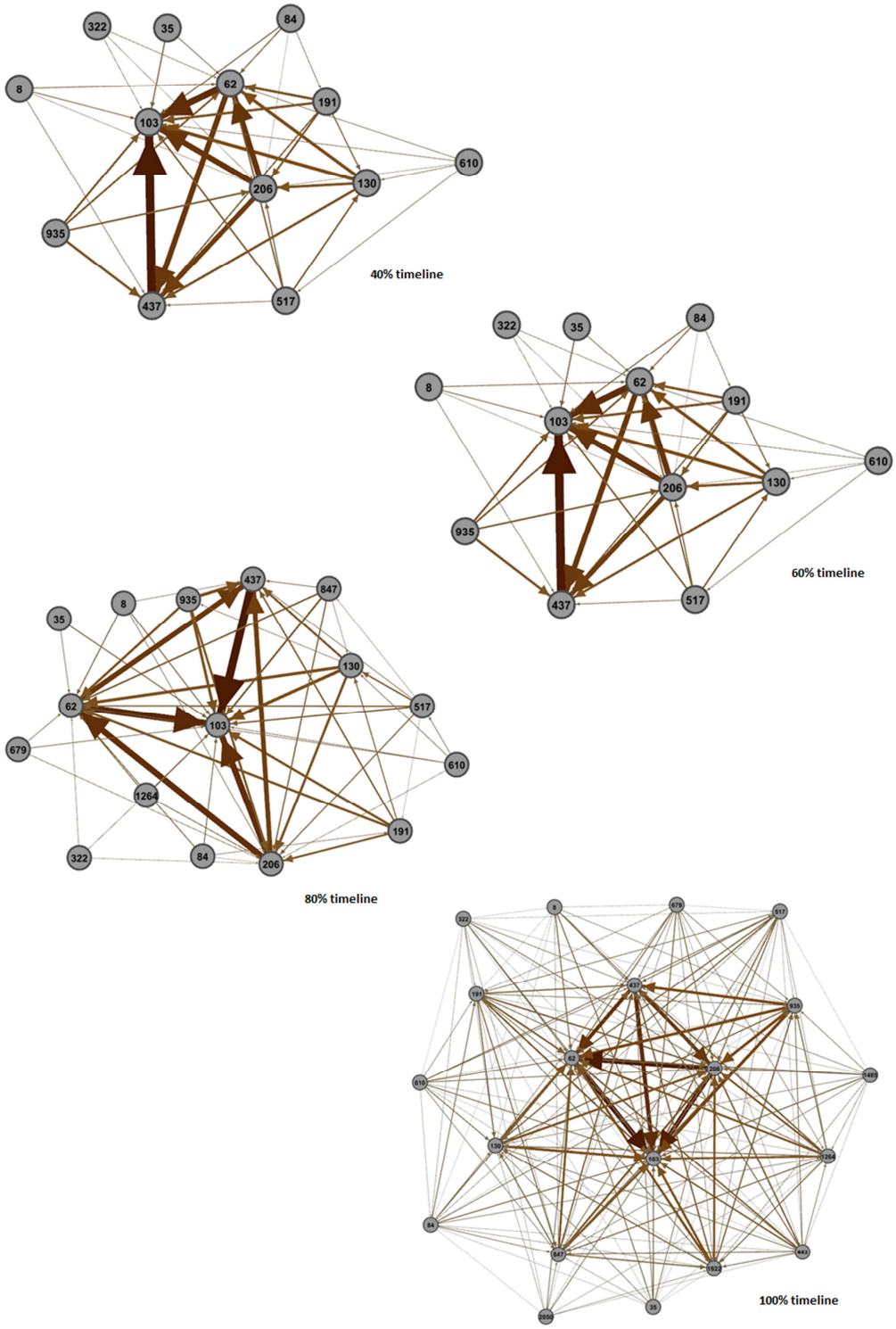

Figura 4.15: Redes formada al 40, 60, 80 y 100% del *timeline* de la red de Acepta el reto, con las aristas de más peso.

Tomando como base los nodos de la red más primitiva (la que está al 40% del *timeline*), se observó la evolución de los pesos de sus aristas (figura 4.15). Y aunque las imágenes son solo una muestra, se puede apreciar como las aristas de más peso y los nodos que conectan, se mantienen a lo largo del tiempo, por lo que siguen estando entre las de más peso de red y según lo observado en los datos, este peso generalmente ha aumentado a lo largo del tiempo.

Algunas aristas asociadas a algunos nodos como el 517 no siguen tal evolución, esto es debido a que ese usuario en concreto dejó de hacer envíos correctos hacia la mitad del *timeline.*

Se interpreta que usuarios con muchos problemas en común tienen el mismo comportamiento, ya que si han resuelto los mismos problemas, siguen coincidiendo en la resolución de otros cuantos.

4.4.6 Conclusiones obtenidas del análisis de la red

Con todo lo visto anteriormente, se pueden extraer algunas conclusiones del análisis de la red social de Acepta el reto, útiles para implantación del algoritmo:

- Las relaciones entre usuarios con un solo problema en común carecen de valor para poder recomendar.
- Un usuario debe tener hechos un mínimo de dos problemas para pedir una recomendación a los demás usuarios.
- Así mismo, un usuario debe tener como mínimo cuatro problemas resueltos para poder recomendar problemas a otro usuario.
- Al pedir una recomendación, aquellos usuarios con mejor relación (más problemas en común con el que hace la petición) son los que pueden hacer la mejor recomendación.

4.5 Creación del algoritmo

A medida que se iban observando los resultados obtenidos del análisis de la red, se empezó a crear el algoritmo y a añadirle criterios a la hora de recomendar problemas. A partir de este proceso, se desarrolló la mecánica definitiva del algoritmo de recomendación.

La lógica del algoritmo está íntimamente relacionada con la red social anteriormente estudiada, donde el usuario que pide una recomendación señala con las aristas a los usuarios relacionados que pueden recomendarle problemas, siendo el peso de las aristas el grado de similitud entre usuarios (figura 4.16).

Inicialmente, al algoritmo se le introduce el usuario consultor, el que desea que se le recomienden problemas. A continuación se consulta el número de problemas que ha resuelto el usuario y dependiendo de si llega al mínimo indicado en la aplicación o no, hará una ejecución u otra.

Si no llega al mínimo de dos problemas resueltos exigido por el algoritmo, devuelve como recomendación una lista de los problemas con más soluciones correctas enviadas, ordenada por el número de envíos correctos y se devolverán al usuario consultor los problemas que estén en las primeras posiciones de esta lista. Una solución tomada del recomendador encontrado en *uHunt*, comentado en la sección 2.1.1 del capítulo Estado del arte.

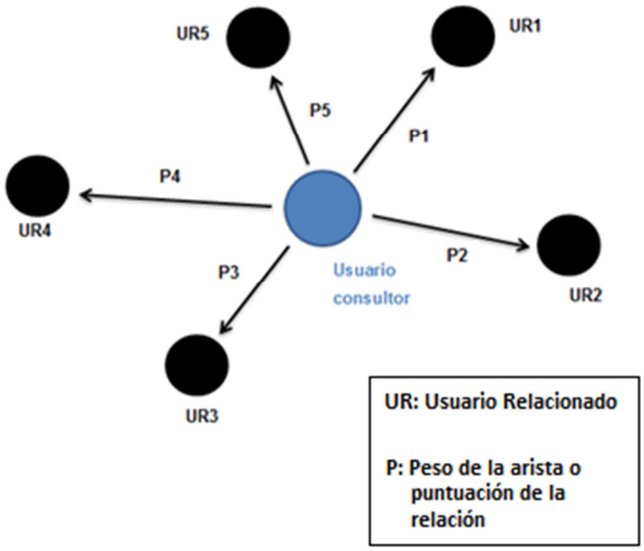

Figura 4.16: Diagrama de las relaciones entre nodos en la red de Acepta el reto.

Si el usuario consultor cumple la condición de tener dos o más problemas resueltos, se utiliza la lógica del diagrama de la figura 4.16: las aristas que salen del usuario consultor, le indican los usuarios relacionados que podrían recomendarle problemas. Los usuarios relacionados tienen P problemas en común con el usuario consultor, además de tener más problemas resueltos en total que el consultor y un mínimo de cuatro problemas resueltos, como ya se explicó en la sección 4.4.5 del capítulo Sistema de recomendación de problemas. El valor P también sirve como medida de puntuación de la fortaleza de la relación, midiendo la similitud del comportamiento entre el usuario consultor con los usuarios relacionados (cuanta más puntuación, más similitud entre ambos usuarios). Con todo lo explicado anteriormente, se crea una lista de la que se seleccionan aquellos usuarios relacionados con más valor de P (aquellos

con más problemas en común con el usuario consultor y que por lo visto en la sección 4.4.6, son aquellos que podrían hacerle una mejor recomendación).

Teniendo ya a los usuarios relacionados que pueden hacer mejores recomendaciones, se localizan aquellos problemas que hayan hecho y que puedan recomendar al usuario consultor (descartando los que este ya haya hecho), creando una lista provisional de recomendación, con los problemas que son recomendados y las veces que son recomendados. El número de veces que es recomendado un problema, se considera como una puntuación de lo recomendable que es un problema (lo adaptado que es para las capacidades del usuario consultor).

De la lista provisional de recomendación creada, se selecciona un número determinado de los problemas que tengan una mayor puntuación (sean más recomendables para el usuario consultor) y se elabora lo que será el resultado de la recomendación: un ranking de los problemas recomendados al usuario, de más recomendable a menos. Por último, este ranking es devuelto al usuario.

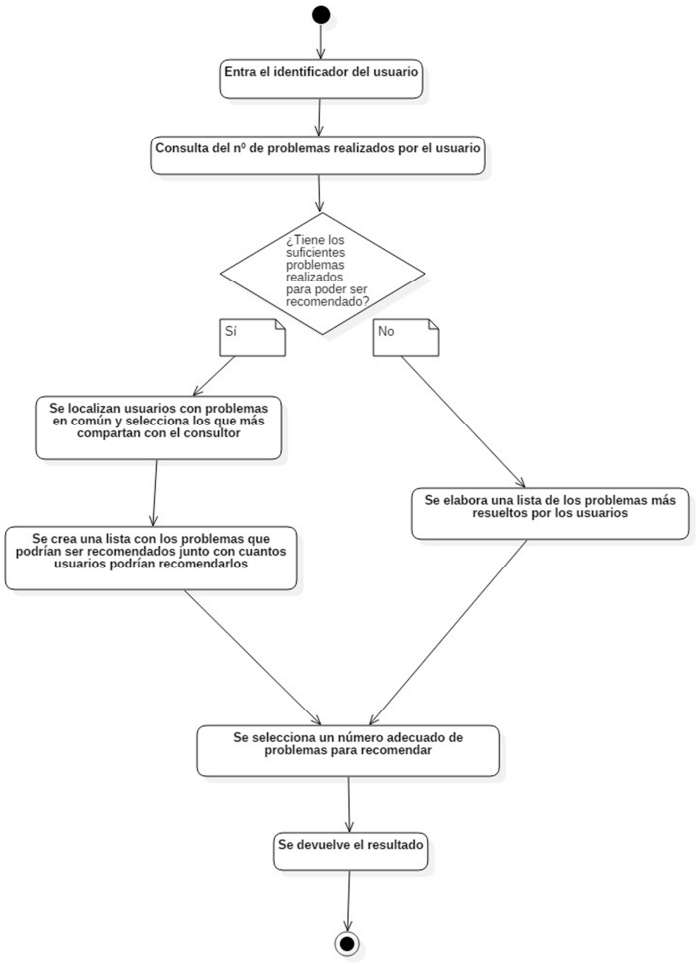

Figura 4.17: Diagrama de actividad del algoritmo de recomendación.

4.6 Elaboración del Servicio Web

Una vez definido el funcionamiento del algoritmo, se comenzó la elaboración del servicio web para el sistema de recomendación de problemas, basado en el comportamiento de los usuarios. Se pidió consejo al profesor responsable de la tutoría del este trabajo, ya que tendría que ser un servicio que se pudiera añadir a la página de Acepta el reto y al menos tendría que coincidir en el lenguaje usado por esta, que es Java. También se decidió que dicho servicio web fuera lo más sencillo posible, ya que su uso sería para probar el algoritmo y su posterior presentación, no para su implantación inmediata en Acepta el reto.

Teniendo esto en cuenta, el lenguaje elegido para su creación fue Java, se usaron las librerías Jersey por la cantidad de documentación existente y su facilidad de uso y se utilizó la arquitectura REST también por la documentación existente y por ser la que está siendo más usada actualmente. También se decidió crear el proyecto sobre la plataforma Maven, específica para proyectos web y con gran capacidad de escalabilidad. Se optó por usar esta plataforma en último caso ya que, aunque se desconocía cómo es el desarrollo real de la página Acepta el Reto, el proyecto podría ser fácilmente exportable, ya que usa una metodología similar a los *frameworks* de desarrollo web más extendidos, además de que buena parte de otros desarrollos observados y tomados como guía para crear el del presente trabajo, estaban también creados sobre esta plataforma.

Para la construcción del servicio web se ha seguido la metodología propia de REST y las buenas prácticas de programación. La estructura interna es la proporcionada por defecto por Maven, incluyendo las clases Java que incluyen el algoritmo de recomendación y el propio servicio que haga las operaciones de recepción y envío de datos.

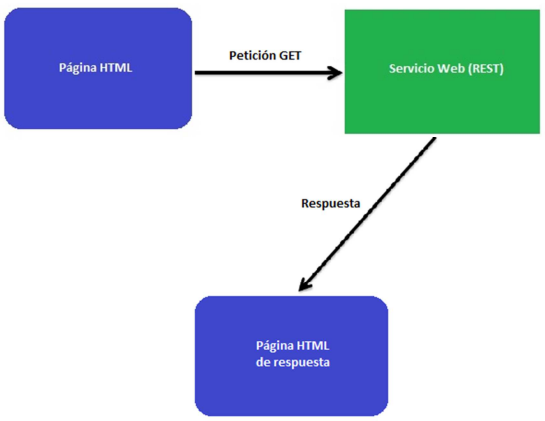

Figura 4.18: Diagrama de funcionamiento del servicio web.

El funcionamiento básico del servicio es el de una petición GET (figura 4.18). Explicado de forma específica, es que al introducir el identificador del usuario, se hace una petición al servidor de los datos que quiere el usuario (en este caso una recomendación de problemas). Esta petición se hace mediante el método HTTP, GET. Es usado este método en concreto porque es el indicado para obtener información del servidor pero sin hacer cambios en él. Los datos obtenidos se devuelven al usuario en formato HTML.

La página mostrada inicialmente por el servicio (figura 4.19) contiene un sencillo formulario en el cual hay que introducir el identificador del usuario al que se quiere hacer la recomendación.

Figura 4.19: Frontal del servicio web.

En el campo proporcionado para introducir el identificador, se controla que el texto introducido solo sea de tipo numérico y mayor a uno. Dicho control se implementa en el mismo elemento HTML, el cual en su versión 5.0 permite realizar estos controles, indicando que sea un objeto “input” de tipo numérico (evitando la introducción de caracteres no numéricos) y con la propiedad “min” igualada a uno (evitando introducir identificadores negativos).

El resto del frontal se compondría de la estructura propia de una página HTML: un título (elemento “h1”) y un botón *submit* que hará la petición GET, además de un elemento “form” que contendrá el campo dedicado a introducir el identificador.

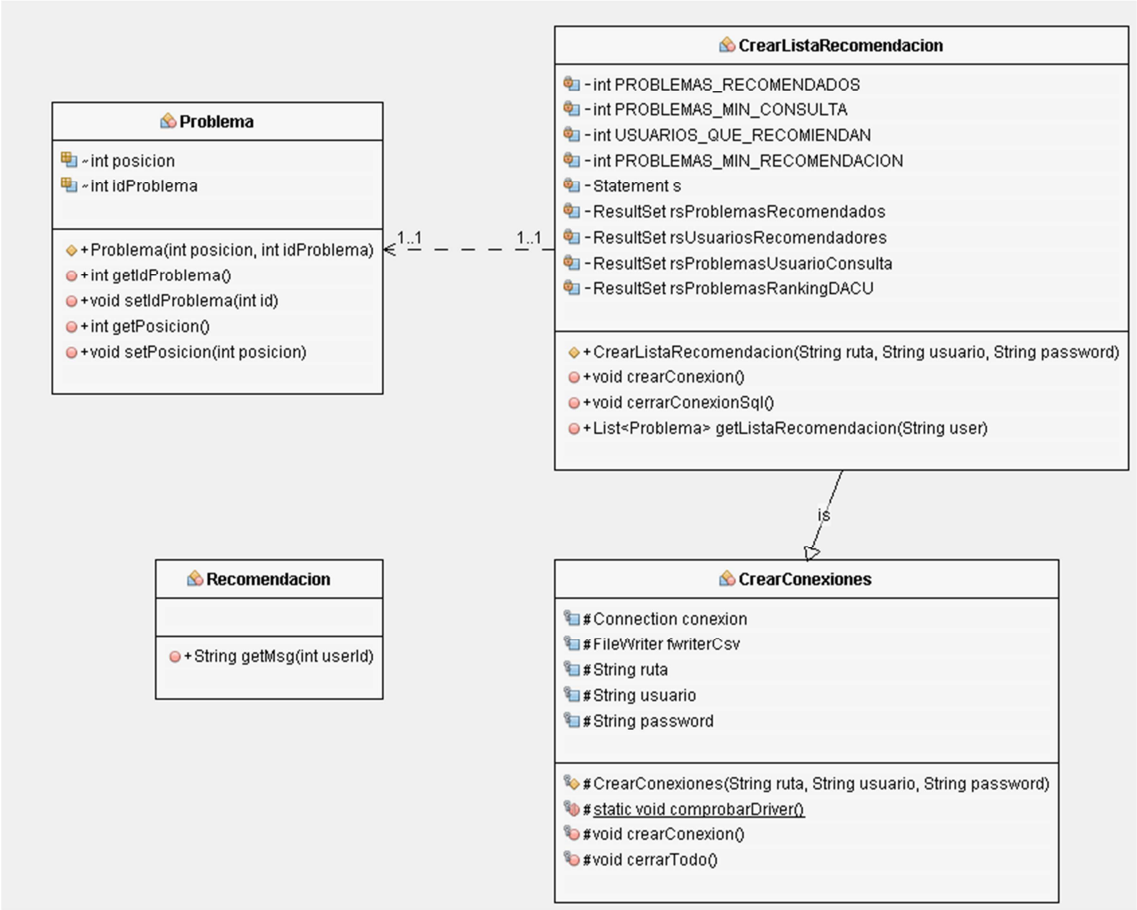

Figura 4.20: Diagrama de clases del servicio web del recomendador.

Las clases usadas por el servicio web son las mostradas en el diagrama superior (figura 4.20). La clase “Recomendacion” es la que contiene el método GET, contiene los datos de la conexión con la base de datos, ejecuta el algoritmo de recomendación y devuelve texto con formato HTML, con el ranking de problemas recomendados. La clase “CrearListaRecomendación” es la que contiene el algoritmo de recomendación y que recibe el identificador del usuario que pide una recomendación y elabora el ranking de problemas recomendados para él. Esta clase hereda de “CrearConexiones”, que contiene los métodos que manejan la conexión con la base de datos. La clase “Problema” sirve como DTO (Data Transfer Object u objeto de transferencia de datos) de los problemas recomendados al final de la ejecución del algoritmo, entre la clase “CrearListaRecomendación” y “Recomendacion”

El funcionamiento consiste en que la clase “Recomendacion” recibe la petición GET y la maneja llamando a la clase “CrearListaRecomendacion”, que prepara el objeto que realizará la recomendación de problemas. Esta clase hereda la clase “CrearConexiones”, que contiene métodos para crear la conexión con la base de

datos) y que le pasa los parámetros de la conexión con la base de datos. Posteriormente se llama al método “getListaRecomendacion” de la clase “CrearListaRecomendación”, instanciada anteriormente, para que devuelva una lista de objetos “Problema”. Los objetos de la clase “Problema” tienen los atributos “posicion” e “idProblema” que serán la posición en el ranking de recomendación e identificador de problema respectivamente. Por último, la clase “Recomendación”, mediante la lista con la información de la recomendación, edita un sencillo texto con formato HTML, que es devuelto y muestra la página con el resultado de la recomendación al usuario, compuesta de un título y una tabla con la recomendación de problemas.

Los parámetros internos usados por el algoritmo son:

- Problemas que serán recomendados en el ranking final devuelto al usuario: 10.
- Problemas que debe tener hechos un usuario para que otros puedan recomendarle problemas: 2.
- Número de usuarios que recomendarán problemas: 20.
- Número mínimo de problemas que debe tener hechos un usuario para poder recomendar problemas: 4.

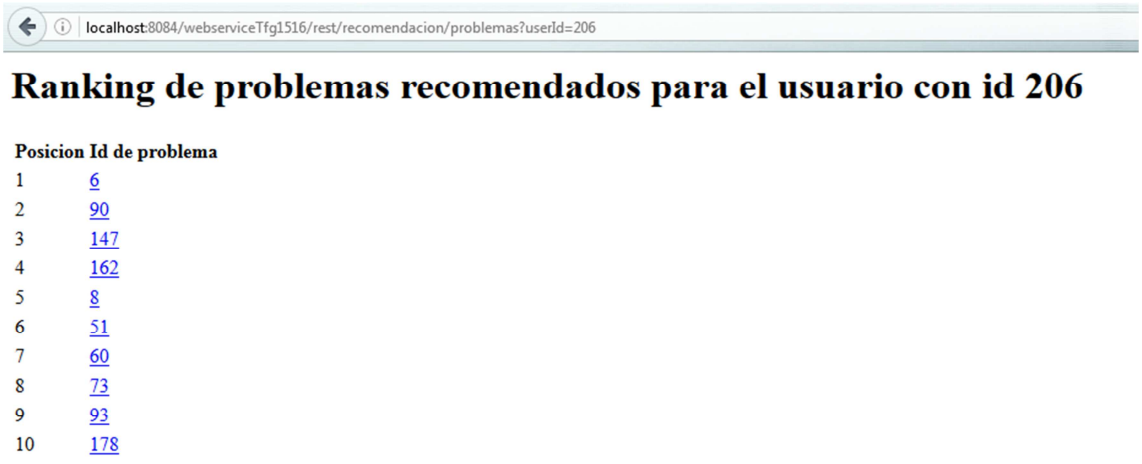

Figura 4.21: Resultado de la recomendación de problemas del servicio web.

El resultado para el usuario, mostrado en la figura 4.21, es una página que esencialmente muestra el ranking de problemas recomendados al usuario, desde la posición 1 con el problema más recomendable hasta la 10 con el problema menos

recomendable de la lista. Además el propio identificador del problema funciona como un enlace al enunciado de dicho problema en la página de Acepta el reto.

5 Validación y pruebas

En este capítulo se hablará de las pruebas realizadas al algoritmo de recomendación para comprobar su funcionamiento y del servicio web que lo alberga

Para las pruebas del algoritmo, se ha dado una nueva funcionalidad al programa auxiliar utilizado anteriormente para crear la red social (sección 4.2.2 del capítulo Sistema de recomendación de problemas) y cuyo manual de la aplicación se encuentra en el anexo A. Esta nueva funcionalidad hace una recomendación de problemas a un usuario en un punto del *timeline* (línea de tiempo de los datos de Acepta el reto) anterior al final y otra recomendación al final del *timeline*. Compara ambos resultados, contando que ejercicios sugeridos en la primera recomendación que han sido resueltos en la última y a partir de esto se crea una tasa de aciertos, siendo aciertos las veces que el algoritmo adivina el comportamiento del usuario, al sugerirle un problema que luego ha terminado haciendo y por tanto acertando el algoritmo en su predicción. Esta tasa se obtiene al dividir los aciertos de la recomendación (problemas de la recomendación que ha resuelto el usuario) entre todos los propuestos en la recomendación, dando una medida de lo eficaz que es el algoritmo recomendando problemas adecuados a los usuarios.

Los usuarios escogidos para las pruebas han tenido las siguientes condiciones:

- Los puntos para realizar la primera recomendación han sido en el 60 y 80 por ciento del *timeline* (figura 5.1). Por lo que habrá dos test sobre el algoritmo.
- Se han elegido aquellos cuyo último envío estaba más cerca del final del *timeline* para así poder comprobar los resultados.
- Los usuarios escogidos tenían envíos antes del punto del 60% del *timeline*, para así poder usar los mismos usuarios en ambos tests y ver la diferencia entre recomendaciones hechas en un punto y otro.
- Todos los usuarios tenían al menos dos envíos correctos antes del punto del 60% del *timeline* para cumplir con las condiciones del algoritmo para recomendar a partir de las relaciones entre usuarios, que es la parte del algoritmo que se quiere probar. El otro tipo de recomendación, basada en el conjunto de envíos totales, no era necesario probarla al estar ya implantada y corroborado su uso (sección 2.1.1 de Estado del arte).
- Se han escogido usuarios que tengan al menos diez envíos correctos al final del *timeline*, para obtener resultados más ricos en datos al crear la oportunidad de que los usuarios pudieran haber resuelto todos los problemas de la recomendación.

- Se ha procurado que aparecieran todos los tipos de usuarios: con pocos envíos al hacer la recomendación, con pocos envíos después de hacer la recomendación y una cantidad equilibrada de envíos antes y después de la recomendación.

En total y según las condiciones antes mencionadas, han sido escogidos 32 usuarios para las pruebas.

Los parámetros internos del algoritmo de recomendación son los mismos que los usados en la implantación en el servicio web, en la sección 4.6 de este capítulo.

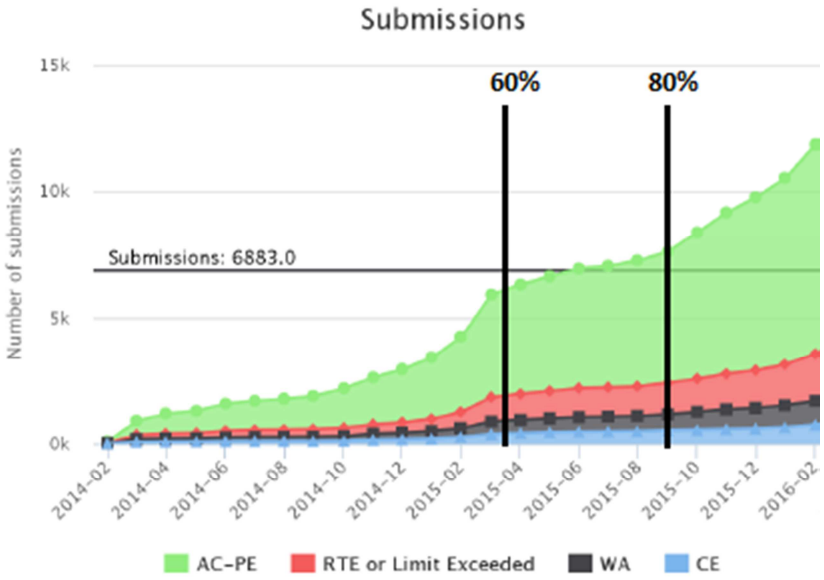

Figura 5.1: Gráfica acumulativa de los envíos hechos a Acepta el reto con los puntos 60% y 80% donde se realizaron las pruebas. (Jiménez Díaz, Gómez Martín, Gómez Martín y Sánchez Ruiz-Granados, 2016).

Los informes de resultados obtenidos, se encuentran en la sección 7.2 del capítulo de Anexos.

- Para las pruebas realizadas en el 60% del *timeline*, se obtiene una tasa de aciertos media del 23,4%.
- Para las pruebas realizadas en el 80% del *timeline*, se obtiene una tasa de aciertos media del 8,4%.

Los resultados de primera vista son prometedores, especialmente los de las pruebas realizadas en el 60% del *timeline*, ya que los usuarios puedieron realizar más problemas para comprobar las recomendaciones, que en los resultados obtenidos en el 80% del *timeline*, en los que los usuarios realizaron menos ejercicios y no se ha

podido comprobar el comportamiento que han seguido a lo largo de mucho tiempo. Se ha considerado que ha habido factores que han impedido obtener mayores tasas de aciertos, ya que con cada nueva resolución de un ejercicio, se crean nuevas relaciones que podrían alterar los resultados de la recomendación y conseguir más aciertos, algo no posible al mantener la recomendación inicial todo el tiempo.

Las pruebas del servicio web se han limitado a comprobar que se implementaba correctamente el algoritmo y a comprobar el sencillo control de errores implementado. Se ha observado que la usabilidad es algo limitada, ya que apenas permite realizar acciones y los datos se reducen a identificadores. Hay que recordar lo dicho en la sección 4.6 del capítulo Sistema recomendador de problemas, que define que la función del servicio web es la de prueba del algoritmo y de que es posible su implantación en Acepta el reto, no un servicio que se pudiera añadir inmediatamente y sin modificaciones. Sobre el uso de identificadores en vez de nombres, es debido a lo explicado en la sección 3.2 del capítulo Punto de partida: Acepta el reto, para cumplir con la protección de datos de los usuarios y en el caso de los nombres de los problema, simplemente porque esos datos no estaban en el extracto de la base de datos utilizado en este trabajo.

6 Conclusiones y líneas futuras

Al término del presente trabajo se puede afirmar que se han cumplido todos los objetivos, ya que se ha creado un algoritmo capaz de recomendar problemas en base a criterios sólidos y ha sido implantado con éxito en un servicio web demostrando sus capacidades.

Las pruebas realizadas demuestran que el algoritmo logra acertar el comportamiento de algunos usuarios y se considera que el algoritmo funciona, adivinando el comportamiento de los usuarios, haciendo recomendaciones adaptadas a cada uno. Además de que también se ha demostrado que puede ser exportado a un servicio web.

Entre los posibles desarrollos futuros basados en este trabajo, está principalmente su integración en la página web Acepta el reto, convirtiéndose en una herramienta útil y que despierte interés entre los usuarios de dicha página.

Un posible desarrollo dentro de la mecánica del algoritmo, sería el de puntuar los problemas recomendados por los usuarios relacionados a partir de la puntuación P que tienen esos usuarios para recomendar (figura 4.16), ya que podría mejorar la tasa de aciertos del algoritmo.

En el desarrollo del servicio web integrado en Acepta el reto, se podría mejorar el rendimiento del actual servicio creado, guardando en memoria caché el mapa de las relaciones entre usuarios (la red social descrita en el capítulo 4) y actualizándolo a medida que se hagan envíos de problemas resueltos correctamente. Así se evitaría hacer pesadas consultas a la base de datos cada vez que un usuario quisiera hacer una petición al servicio.

Otro posible proyecto basado en el actual, podría ser el de un sistema de emparejamiento de usuarios o búsqueda de posibles contrincantes en campeonatos, ya que la base del algoritmo se basa en la búsqueda de coincidencias entre usuarios en base a su comportamiento y sería adecuado para este fin, además de que el desarrollo adicional sobre la aplicación sería mínimo.

Por último el propio estudio sobre las relaciones entre usuarios basados en su comportamiento y el proceso seguido, puede ser usado como base para futuros trabajos, tanto en el ámbito educativo como en otros entornos.

Traducciones

A continuación están las traducciones de los capítulos de Introducción y Conclusiones y líneas futuras.

7 Introduction

This paper deals with the creation of a social recommendation model in a web teaching environment, specifically for *Acepta el reto* website. The purpose of this work is the development of an exercise recommendation algorithm based on user's behavior and its implementation in a web service that can be used by this website.

7.1 Motivation

It has been chosen this topic for the existing problem in those who decide to solve exercises voluntarily to implement their skills, in particular through an online judge: a web system to test programming code, which is also used in programming exercises practice, and it is issuing verdicts on the submitted solutions.

Because there are a large number of possible problems to solve, it is easy for those who decide to practice exercises, fall into indecision or a bad choice, as a problem too difficult for their abilities or uninteresting and end up leaving.

Specifically, this case can be found on *Acepta el reto* website, which houses an extensive collection of programming problems and an online judge passing verdicts on the collection of sent solutions. On that page users can choose to solve any problem, but this choice may lead to cases in which an inexperienced user choose a problem that initially seems interesting but too difficult for him, abandoning the website and possibly new attempts to practice exercises voluntarily. Otherwise it could happen that an experienced user chooses an extremely easy problem to solve, with similar result.

7.2 Goals

Work goals are:

- Search in pages that contains an online judge dedicated and a collection of programming exercises, if there is a problem referral service.
- Use social network analysis methods to find out how to recommend problems to users.
- Create a problem recommendation algorithm for users of *Acepta el reto* website, only with the information provided by his database.
- Validate performance of this algorithm.

- Implement this algorithm in a web service to try and show how it can be added to *Acepta el reto* website.

7.3 Work plan

Initially the working team for this study was two people, but one of the members left the project before starting its development part of it, so in practice the project was carried out by a single member.

Before the previously commented abandonment, it decided coordination team, opting for a Scrum type methodology because many changes were expected from the original idea and it could be adapted well to this project. This agile methodology was made to have partial deliveries in regular periods (defined in two weeks) to give a vision of development and could be used to improve aspects of final delivery, besides that continually have to modify the algorithm to improve efficiency. Initially for member's communication, online messaging system was adopted, apart from scheduled meetings.

However, after the abandonment named above, the methodology defined above became useless, although it was adapted by conducting regular reviews, also every two weeks, of all work done, for while going writing what would be this paper .

To development, a repository on Google Drive was enabled, consisting of documentation folders, code and various elements that might be useful. Was adopted a modification date model for versioning.

To prepare the project and the drafting of the paper, it has been used mainly Internet documentation, both explanatory documents on used technologies as tools technical documentation necessary to make this work.

7.4 Used technologies

Used machine to develop, was a PC with an Intel i5 processor at 2.8GHz with 12GB of RAM and using Windows 7 operating system as a development platform.

Code writing language was Java in version 1.8, using Eclipse IDE and NetBeans, in addition to the libraries provided by default, Jersey libraries were used for the development of web services and MySql libraries for database connection. The database management system was MySql and the database management application was MySQL Workbench. The Web service project was made on Maven, using for testing Firefox web browser.

7.5 Document structure

The current paper is intended to be an explanation of the work performed, exposing the taken steps since its inception to completion, commenting entire process.

It has an introduction where the motivations and goals of the work are explained, as well as the methodology used in its preparation.

After, State of the art section is exposed, where knowledge used in job creation are explained and used to a better understanding of it.

The following describes *Acepta el reto* website, the beginning of this work and its base.

Next chapter is the explanation of the recommendation system. Includes processing, with a study of the created social network, performed to determine the recommendation algorithm structure, the algorithm creation and the web service that houses it.

The next chapter details the testing and validation of the created recommendation algorithm.

Finally the conclusions that have been reached and future lines of this work are discussed.

8 Conclussions and future lines

At the end of this work we can say that they have met all objectives, as it has created an algorithm able to recommend problems based on solid criteria and has been successfully implemented in a web service demonstrating their capabilities.

Tests show that the algorithm hit some users' behavior and it is considered that the algorithm works, divining user's behavior, making tailored recommendations to each. Besides that also been shown that it can be exported to a web service.

Among possible future developments based on this work it is mainly their integration into *Acepta el reto* web page, becoming a useful tool to arouse interest among users of this page.

A possible development within algorithm mechanics would be rate recommended problems related from the P score that those users have for to recommend (Figure 4.16), as it could improve the hit rate of algorithm users.

In the development of the integrated *Acepta el reto* web service, it could improve performance of current service created, keeping in cache relationships map between users (social network described in Chapter 4) and updating it as problems solved correctly shipments are made. This would avoid doing heavy queries to database each time a user wants to make a service request.

Another possible project based on the current, could be a user matching system or a potential tournament opponents search because the basis of the algorithm is based on the search for matches between users based on their behavior and would be suitable for this purpose, in addition to the further development of the application would be minimal.

Finally itself study based on the users behavior relationship and the process followed, can be used as a basis for future work, both in education and in other environments.

9 Referencias

- Wikipedia. (2016). Gephi. [en línea] Disponible: https://en.wikipedia.org/wiki/Gephi [Acceso Ago. 2016].
- Es.wikipedia.org. (2016) a. *Sistema de recomendación*. [en línea] Disponible: https://es.wikipedia.org/wiki/Sistema_de_recomendación [Acceso Ago. 2016].
- Jarroba. (2013) a. *¿Qué son los Sistemas de Recomendación? - Jarroba*. [en línea] Disponible: http://jarroba.com/que-son-los-sistemas-de-recomendacion/ [Acceso Ago. 2016].
- Jarroba. (2013) b. *Sistemas de Recomendación basados en Filtrado Colaborativo (K-Vecinos) - Jarroba*. [en línea] Disponible: http://jarroba.com/sistemas-de-recomendacion-basados-en-filtrado-colaborativo-k-vecinos/ [Acceso Ago. 2016].
- Es.wikipedia.org. (2016) b. *Servicio web*. [en línea] Disponible: https://es.wikipedia.org/wiki/Servicio_web [Acceso Ago. 2016].
- Es.wikipedia.org. (2016) c. *Java Servlet*. [en línea] Disponible: https://es.wikipedia.org/wiki/Java_Servlet [Acceso Ago. 2016].
- Es.wikipedia.org. (2016) d. *Representational State Transfer*. [en línea] Disponible: https://es.wikipedia.org/wiki/Representational_State_Transfer [Acceso Ago. 2016].
- Docs.oracle.com. (2016). *Java Platform SE 8*. [en línea] Disponible: https://docs.oracle.com/javase/8/docs/api/ [Acceso Ago. 2016].
- Jersey.java.net. (2016). Jersey 2.23.2 User Guide. [en línea] Disponible: https://jersey.java.net/documentation/latest/ [Acceso Ago. 2016].
- Redmond, J. (2016). Maven – Maven Documentation. [en línea] Maven.apache.org. Disponible: https://maven.apache.org/guides/ [Acceso Ago. 2016].
- Gephi.org. (2016). Learn how to use Gephi. [en línea] Disponible: https://gephi.org/users/ [Acceso Ago. 2016].
- Jiménez Díaz, Guillermo y Díaz, Alberto (2016). *Apuntes de Análisis de Redes Sociales*. Profesores. Universidad Complutense de Madrid.
- Jiménez Díaz, G., Gómez Martín, P., Gómez Martín, M. y Sánchez Ruiz-Granados, A. (2016). *Similarity Metrics from Social Network Analysis for Content Recommender Systems*. Universidad Complutense de Madrid.

Anexo A: Manual de "AppAuxiliarGephi"

Este apartado trata de la aplicación auxiliar que genera los archivos que serán exportados a Gephi para el estudio de las redes y de generar los informes de pruebas del algoritmo de recomendación.

La aplicación es entregada en formato JAR con las librerías MySql para la conexión con la base de datos, además de entregarse el código fuente.

A.1 Diagrama de clases de la aplicación

A continuación se muestra el diagrama de clases de la aplicación "AppAuxiliarGephi".

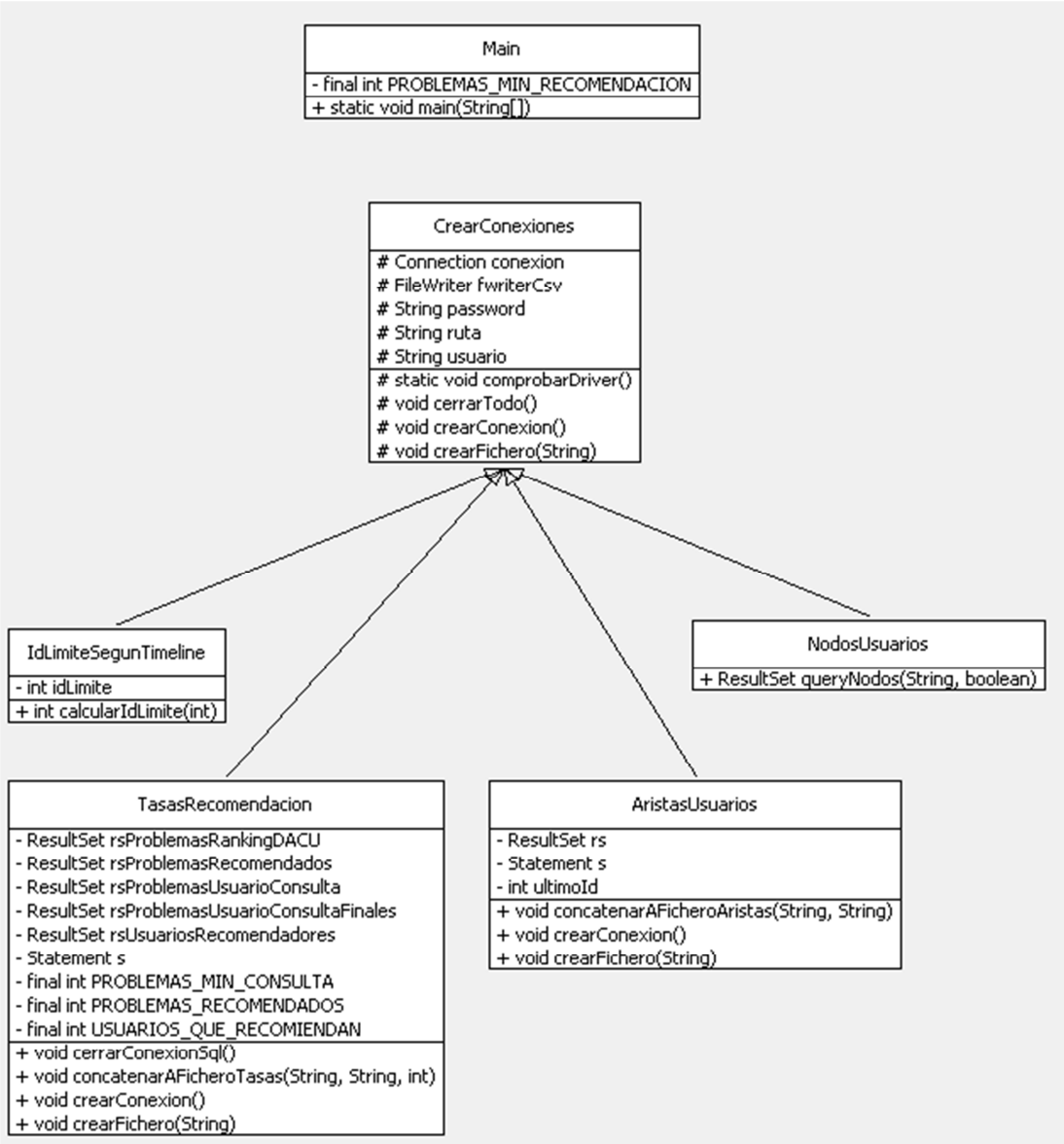

Figura A.1: Diagrama de clases de "AppAuxiliarGephi", aplicación auxiliar.

La clase “Main” es la que contiene los datos de conexión con la base de datos y muestra el menú con las opciones de ejecución, además de dirigir la ejecución de la aplicación, llamando al resto de clases.

La clase “CrearConexiones” es la encargada de crear la conexión con la base de datos y el resto de clases heredan sus métodos para dicha conexión.

La clase “NodosUsuarios” es la encargada de buscar los usuarios que han resuelto problemas en Acepta el reto y crear el archivo de nodos para la red social. La función de “AristasUsuarios” es similar sólo que crea el archivo de las aristas, pudiendo crearse las aristas existentes hasta cierto punto del *timeline*.

La clase “IdLimiteSegunTimeline” tiene la única función de devolver el identificador de usuario límite para el punto del *timeline* que se le indique.

Por último, la clase “TasasRecomendacion” es la que contiene una implantación del algoritmo de recomendación y dirige la ejecución de pruebas sobre este, creando los informes de pruebas existentes en el anexo B.

A.2 Ejecución de la aplicación

Esta aplicación está comprimida en un archivo de formato JAR y funciona sobre consola, por lo que en Windows será necesario ejecutarlo desde Símbolo del Sistema o cmd.

Una vez abierto el cmd, apuntar al directorio donde se encuentra “AppAuxiliarGephi.jar” y escribir en la línea de comandos:

```
C:\Directorio>java -jar AppAuxiliarGephi.jar
```

Una vez ejecutada mostrará una serie de opciones con las funcionalidades de la aplicación. Hay que introducir el número de la opción que se desea ejecutar para que la aplicación la ponga en funcionamiento. Al introducir el número de la opción, aparecerá la frase “procesando…”, que indica que se está ejecutando la operación indicada anteriormente Una vez terminada la ejecución de la opción elegida, aparecerá la palabra “terminado” y volverá a aparecer el mismo menú.

A.3 Conexión con la BD

Esta aplicación debe poder conectarse con una base de datos MySql que contenga la estructura de la base de datos de Acepta el reto. En caso de usar la aplicación contenida en el archivo “jar”, para que la conexión sea correcta la base de datos debe estar en el servidor local (*localhost*) y el esquema donde esté llamarse “tfg”, para así

coincidir con la ruta usada por la aplicación: “127.0.0.1:3306/tfg“, así mismo el usuario utilizado es “root” y la contraseña “1234”.

En el caso de que la base de datos tenga definida otra configuración, se deberá cambiar el código fuente, concretamente la clase “Main” y compilar de nueva la aplicación.

A.4 Crear archivo de todos los nodos

Para crear el archivo de nodos, de las opciones proporcionadas por la aplicación, hay que elegir: “0 Crear los nodos de los usuarios”. Al terminar de ejecutarse en el directorio donde está la aplicación, se habrá creado el archivo “NodosUsuarios.csv” que contiene los datos de los nodos de todos los usuarios, listos para ser importados a Gephi.

A.5 Crear archivo de todas las aristas

Para crear el archivo de todas las aristas (todas las relaciones entre usuarios existentes en la base de datos), hay que elegir la opción: “1 Aristas de usuarios relacionados por los problemas enviados y de veredicto AC”. Al terminar de ejecutarse en el directorio donde está la aplicación, se habrá creado el archivo “AristasUsuariosTodas.csv” que contiene los datos de los nodos de todos los usuarios, listos para ser importados a Gephi.

A.6 Crear archivo de aristas hasta porcentaje de *timeline*

Para crear el archivo de todas las aristas hasta cierto porcentaje del *timeline* total, hay que elegir la opción: “2 Aristas de usuarios relacionados limitado por timeline” y se mostrará: “Introduzca el porcentaje del timeline a calcular:” teniendo que introducir a continuación hasta que porcentaje de *timeline* se quieren calcular las aristas. Al terminar de ejecutarse en el directorio donde está la aplicación, se habrá creado el archivo “AristasUsuariosXX%Timeline.csv”, donde XX es el porcentaje introducido de *timeline*. Dicho archivo está listo para importarse a Gephi.

A.7 Crear informe de pruebas del algoritmo

Para crear el informe de pruebas realizadas sobre el algoritmo, hay que elegir la opción: “3 Comprobacion de resultados por usuarios y porcentaje de timeline”, mostrándose a continuación: “Introduzca en qué porcentaje de timeline realizar la consulta”, dejando al usuario introducir el momento, en porcentaje del *timeline*, en el que realizar la consulta. Una vez introducido mostrará a modo informativo: “Se usaran datos hasta id de envio =XX” donde XX es el identificador límite hasta donde tomará datos. Justo después muestra: “A continuacion introduzca los id de los usuarios a

consultar. Para terminar escriba 'fin'", teniendo después que escribir los identificadores de usuario en los que hacer la consulta, escribiendo uno por línea (y saltando entre líneas con Enter) y al terminar de introducir identificadores, en una última línea escribir "fin" para terminar la introducción de datos. Después de esto aparecerá la frase "procesando…" y por último, al terminar la creación del informe, "terminado". El archivo originado, como archivo de texto sin formato, se encontrará en el mismo directorio donde estuviere la aplicación, con el nombre "TasasRecomendacionXX%timeline.rtf", donde XX es el porcentaje de *timeline* introducido anteriormente.

A.8 Finalizar la ejecución

Para terminar la ejecución, se debe elegir la opción: "9 Salir", escribiendo la cifra 9.

Anexo B: Informes de pruebas del algoritmo

Una de las opciones de la aplicación auxiliar es la de generar informes con las pruebas de comportamiento del algoritmo, tomando como punto de partida un porcentaje del *timeline* total de los datos y generando en ese punto una recomendación a un usuario. Después compara la recomendación hecho con el comportamiento que ha seguido el usuario en el resto del *timeline* y mostrando en el informe, por cada usuario:

- Identificador el usuario.
- Problemas recomendados.
- Tasa de aciertos como problemas acertados / problemas recomendados.
- Problemas resueltos restando los contados como acertados por la recomendación.

Al principio de cada informe aparece un pequeño texto explicativo de lo que incluye.

B.1 Tasa de recomendación en el 60% del *timeline*

```
A continuacion aparecen los usuarios que realizaron una
consulta en el punto indicado del timeline y los problemas
acertados a partir de esa recomendacion como tasa de aciertos
sobre recomendaciones y aquellos que hicieron fuera de la
recomendacion

Usuario: 840
Problemas recomendados: 134 136 159 109 183 33 139 191 13 29
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 31

Usuario: 935
Problemas recomendados: 4 15 19 23 29 35 39 49 53 78
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 129

Usuario: 1504
Problemas recomendados: 39 13 44 6 29 62 105 49 51 4
Tasa de aciertos: 5/10 | Problemas resueltos restantes: 14

Usuario: 62
Problemas recomendados: 8 10 17 27 57 65 76 86 93 114
Tasa de aciertos: 8/10 | Problemas resueltos restantes: 148

Usuario: 206
Problemas recomendados: 6 73 90 147 162 178 8 10 17 27
Tasa de aciertos: 3/10 | Problemas resueltos restantes: 150

Usuario: 511
Problemas recomendados: 39 62 105 109 15 29 44 100 254 259
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 10

Usuario: 33
Problemas recomendados: 13 33 134 109 29 62 139 159 187 203
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 9
```

```
Usuario: 25
Problemas recomendados: 256 39 259 134 2 136 203 209 257 29
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 23

Usuario: 1584
Problemas recomendados: 2 13 44 62 109 29 33 49 51 100
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 19

Usuario: 431
Problemas recomendados: 256 2 44 141 254 134 13 109 241 29
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12

Usuario: 847
Problemas recomendados: 105 203 209 310 312 35 49 119 143 145
Tasa de aciertos: 4/10 | Problemas resueltos restantes: 86

Usuario: 1460
Problemas recomendados: 39 62 2 49 117 134 159 191 388 29
Tasa de aciertos: 6/10 | Problemas resueltos restantes: 18

Usuario: 979
Problemas recomendados: 109 254 2 29 44 49 134 155 256 13
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 15

Usuario: 854
Problemas recomendados: 39 109 44 2 49 183 327 254 258 390
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 1

Usuario: 437
Problemas recomendados: 13 25 33 44 47 51 60 70 81 95
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 146

Usuario: 689
Problemas recomendados: 203 39 62 141 2 105 134 159 187 310
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 16

Usuario: 47
Problemas recomendados: 33 29 136 256 258 183 209 62 139 159
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 15

Usuario: 254
Problemas recomendados: 39 134 109 251 253 33 136 241 257 13
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 18

Usuario: 695
Problemas recomendados: 109 105 2 39 100 117 203 256 29 114
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 8

Usuario: 1190
Problemas recomendados: 258 259 256 39 134 203 255 109 136
209
Tasa de aciertos: 6/10 | Problemas resueltos restantes: 37

Usuario: 105
Problemas recomendados: 39 109 209 134 155 183 187 203 33 136
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 9
```

```
Usuario: 1037
Problemas recomendados: 256 39 203 253 259 134 49 187 209 136
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12

Usuario: 1480
Problemas recomendados: 134 191 203 141 159 62 109 136 183 33
Tasa de aciertos: 3/10 | Problemas resueltos restantes: 17

Usuario: 382
Problemas recomendados: 109 39 105 62 159 251 390 2 128 134
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 9

Usuario: 443
Problemas recomendados: 44 2 13 49 51 60 134 29 33 15
Tasa de aciertos: 3/10 | Problemas resueltos restantes: 50

Usuario: 89
Problemas recomendados: 13 15 39 62 109 33 44 105 134 136
Tasa de aciertos: 10/10 | Problemas resueltos restantes: 27

Usuario: 1531
Problemas recomendados: 109 105 114 119 2 39 316 336 342 346
Tasa de aciertos: 3/10 | Problemas resueltos restantes: 6

Usuario: 1317
Problemas recomendados: 44 6 13 134 109 258 49 70 136 141
Tasa de aciertos: 3/10 | Problemas resueltos restantes: 11

Usuario: 1475
Problemas recomendados: 39 134 150 159 183 187 136 33 203 109
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 8

Usuario: 1264
Problemas recomendados: 62 100 13 29 105 203 209 15 49 51
Tasa de aciertos: 4/10 | Problemas resueltos restantes: 59

Usuario: 686
Problemas recomendados: 134 187 159 109 183 203 155 33 136
150
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 24

Usuario: 871
Problemas recomendados: 39 254 109 256 258 134 155 241 259 2
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 7
```

B.2 Tasa de recomendación en el 80% del *timeline*

```
A continuacion aparecen los usuarios que realizaron una
consulta en el punto indicado del timeline y los problemas
acertados a partir de esa recomendacion como tasa de aciertos
sobre recomendaciones y aquellos que hicieron fuera de la
recomendacion

Usuario: 840
Problemas recomendados: 134 109 139 159 44 136 141 183 13 49
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 31
```

```
Usuario: 935
Problemas recomendados: 4 15 17 19 23 29 35 39 49 53
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 131

Usuario: 1504
Problemas recomendados: 44 134 155 159 139 209 141 162 62 136
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 22

Usuario: 62
Problemas recomendados: 8 25 47 93 124 250 264 311 340 358
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 164

Usuario: 206
Problemas recomendados: 6 51 60 73 90 147 162 178 8 25
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 156

Usuario: 511
Problemas recomendados: 39 109 15 27 29 33 51 62 105 254
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 10

Usuario: 33
Problemas recomendados: 13 33 109 15 62 134 254 327 29 49
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 11

Usuario: 25
Problemas recomendados: 39 2 109 183 33 256 259 441 44 49
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 23

Usuario: 1584
Problemas recomendados: 139 187 134 44 109 141 183 2 136 209
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 21

Usuario: 431
Problemas recomendados: 256 2 141 44 134 254 13 33 109 209
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12

Usuario: 847
Problemas recomendados: 119 143 309 310 312 388 503 10 49 57
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 90

Usuario: 1460
Problemas recomendados: 2 39 62 209 4 10 44 134 159 187
Tasa de aciertos: 5/10 | Problemas resueltos restantes: 20

Usuario: 979
Problemas recomendados: 109 155 254 134 2 44 141 150 187 209
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 17

Usuario: 864
Problemas recomendados: 2 134 13 33 187 258 15 29 49 141
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 20

Usuario: 437
Problemas recomendados: 13 33 44 70 81 95 136 150 253 325
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 146
```

```
Usuario: 689
Problemas recomendados: 39 2 13 29 33 134 155 159 162 183
Tasa de aciertos: 2/10 | Problemas resueltos restantes: 14

Usuario: 47
Problemas recomendados: 33 139 162 29 136 150 159 183 209 35
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 17

Usuario: 254
Problemas recomendados: 39 109 134 44 49 183 187 13 136 139
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 20

Usuario: 695
Problemas recomendados: 109 39 105 2 114 117 10 13 29 100
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 8

Usuario: 1190
Problemas recomendados: 258 256 259 39 134 109 209 307 44 141
Tasa de aciertos: 4/10 | Problemas resueltos restantes: 41

Usuario: 105
Problemas recomendados: 2 209 254 39 105 258 307 471 49 62
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 11

Usuario: 1037
Problemas recomendados: 39 134 256 209 253 259 309 49 141 159
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12

Usuario: 1480
Problemas recomendados: 109 139 2 141 155 159 183 209 33 44
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 23

Usuario: 382
Problemas recomendados: 109 39 105 128 254 159 390 2 62 134
Tasa de aciertos: 1/10 | Problemas resueltos restantes: 9

Usuario: 443
Problemas recomendados: 33 136 191 259 465 4 27 49 51 105
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 56

Usuario: 89
Problemas recomendados: 109 49 134 155 159 187 254 100 136
139
Tasa de aciertos: 6/10 | Problemas resueltos restantes: 35

Usuario: 1531
Problemas recomendados: 109 2 39 134 49 70 183 13 33 44
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12

Usuario: 1317
Problemas recomendados: 139 183 13 136 141 150 155 187 209
254
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 17

Usuario: 1475
Problemas recomendados: 39 49 109 2 159 33 139 183 187 29
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 12
```

```
Usuario: 1264
Problemas recomendados: 27 209 62 100 119 143 171 183 217 233
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 67

Usuario: 686
Problemas recomendados: 134 109 187 139 203 209 33 136 155
159
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 24

Usuario: 871
Problemas recomendados: 254 39 109 256 258 134 155 209 2 49
Tasa de aciertos: 0/10 | Problemas resueltos restantes: 7
```